DiffAssemble: A Unified Graph-Diffusion Model for 2D and 3D Reassembly

Gianluca Scarpellini* Stefano Fiorini* Francesco Giuliari* Pietro Morerio Alessio Del Bue
Pattern Analysis and Computer Vision (PAVIS)
Istituto Italiano di Tecnologia (IIT)
Equal contributions

Abstract

Reassembly tasks play a fundamental role in many fields and multiple approaches exist to solve specific reassembly problems. In this context, we posit that a general unified model can effectively address them all, irrespective of the input data type (images, 3D, etc.). We introduce ***DiffAssemble****, a Graph Neural Network (GNN)-based architecture that learns to solve reassembly tasks using a diffusion model formulation. Our method treats the elements of a set, whether pieces of 2D patch or 3D object fragments, as nodes of a spatial graph. Training is performed by introducing noise into the position and rotation of the elements and iteratively denoising them to reconstruct the coherent initial pose.* ***DiffAssemble*** *achieves state-of-the-art (SOTA) results in most 2D and 3D reassembly tasks and is the first learning-based approach that solves 2D puzzles for both rotation and translation. Furthermore, we highlight its remarkable reduction in run-time, performing 11 times faster than the quickest optimization-based method for puzzle solving. Code available at* `https://github.com/IIT-PAVIS/DiffAssemble`.

1. Introduction

Spatial Intelligence is the ability to perceive the visual-spatial world accurately and to perform transformations upon the perceived space [16]. This skill is commonly assessed with reassembly tasks, which involve arranging and connecting individual components to form a coherent and functional entity. Examples of such tasks include solving 2D jigsaw puzzles or assembling 3D objects with LEGO blocks. Since the proposal of the first puzzle solver [13], *Spatial Intelligence* has challenged the Machine Learning (ML) community with its intrinsic combinatorial complexity and its numerous applications, such as genomics [32], assistive technologies [53], fresco reconstruction [2, 48] and molecular docking [7].

Reassembling a set involves placing each element in its correct position and orientation to form a coherent structure, that being a 2D jigsaw puzzle or a 3D object, as in Figure 1.

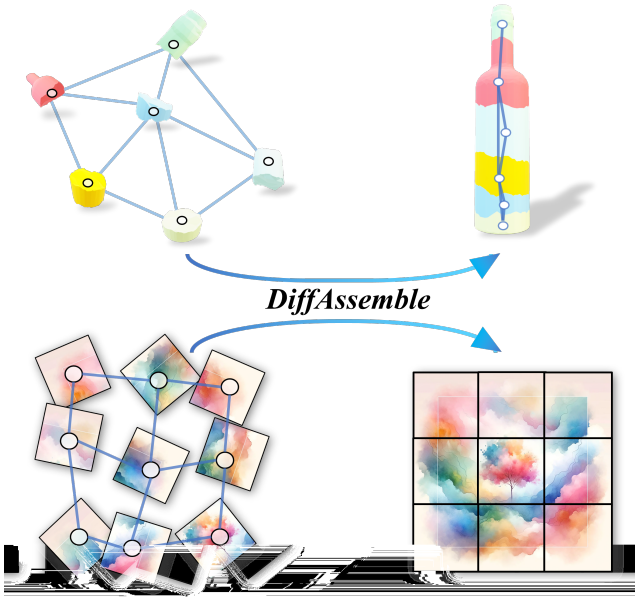

Figure 1. We propose **DiffAssemble** as a unified approach to deal with reassembly tasks in two and three dimensions. DiffAssemble processes the elements to reassemble as a graph and infer their correct position and orientation in 2D and 3D space.

Despite the similarities between the tasks, the literature addresses reassembly tasks in different dimensions separately.

In the 2D dimension, the most common reassembly problem is related to the resolution of puzzles, particularly those with pieces that are translated and rotated and have a regular shape, i.e., square pieces of the same dimension. Due to the regularity of the pieces, the problem can be treated as a permutation problem and solved via optimization-based approaches [15, 22, 52]. These solutions are effective but lack robustness, showing a massive drop in performance when dealing with non-standard scenarios, such as eroded or missing pieces [44]. On the other hand, recent learning-based solutions are robust to distortion in the visual aspect of the pieces by working in the feature space but cannot handle rotations and perform worse than greedy approaches in the standard scenario.

Regarding the 3D reassembly task, since the 3D pieces are not regular, it can not be solved as a permutation problem but has to be solved in the continuous domain, making it a much more challenging task where optimization-based solutions cannot be applied. As a part of the ongoing efforts to address fractured 3D object reassembly, Sellan *et al*. recently introduced the *Breaking Bad* dataset [37] that includes fragments of thousands of 3D objects, and it is commonly adopted as a benchmark for 3D object reassembly solutions. Despite the interest of the machine learning community, the results achieved in 3D reassembly tasks have yet to reach the same level of performance as their 2D counterparts due to the increased complexity of the task.

We argue that 2D jigsaws and 3D objects are two aspects of the same problem, namely reassembly. All these tasks share some properties and, potentially, common solutions. Nonetheless, methods that tackle only one of these tasks are too narrow to generalize to the others. A unique approach that tackles all reassembly tasks at once may benefit from their shared characteristics.

This work introduces **DiffAssemble**, a general framework for solving reassembly tasks using graph representations and a diffusion model formulation. In contrast to prior learning-based approaches for reassembly tasks, which typically tackle the problem in a single step, our approach uses a multi-step solution strategy leveraging Diffusion Probabilistic Models (DPMs) to guide the process. We represent the elements to be reassembled using a graph formulation, allowing us to work with an arbitrary number of pieces. Each piece is modeled as a node that contains the piece's visual appearance, extracted with an equivariant encoder, and the piece's position and orientation. By mapping the appearance to a latent space, we can remove the separation that exists between 2D and 3D tasks and propose a unique solution.

We structure the learning problem using the Diffusion Probabilistic Models (DPM) formulation. We iteratively add Gaussian Noise to each piece's starting position and orientation until they are randomly placed in the Euclidean space. We then train an Attention-based Graph Neural Network [40] to reverse this noising process and retrieve the pieces's original pose from a random starting position and orientation. By adopting a sparsifying mechanism [41] on the graph, we run DiffAssemble on graphs with up to 900 nodes with minimal loss in accuracy while greatly reducing the memory requirement.

Our solution achieves state-of-the-art performance in most 2D and 3D tasks, showcasing that these tasks share common characteristics and can thus be solved through a unified approach. In *2D,* compared to optimization-based solutions, our solution is more robust to missing pieces and much faster, i.e., 5 seconds to rearrange 900 pieces compared to 55 seconds for the fastest optimization approach. In *3D,* our method achieves state-of-the-art results in both rotation and translation accuracy without sacrificing one for the other, as is the case for previous learning-based solutions.

Main Contributions and Novelty of the Work:

- We propose **DiffAssemble**, a unified learning-based solution using diffusion models and graphs neural networks for reassembly tasks that achieve SOTA results in most 2D and 3D without distinguishing between the two.
- We show that reassembly tasks in 2D and 3D share several key properties and that model choices such as the use of different losses, different diffusion chains, and equivariant features.
- To the best of our knowledge, DiffAssemble is the first learning-based solution that can handle rotations and translations for 2D visual puzzles.

2. Related Works

In this section, we revise the main literature for reassembly tasks in 2D and 3D. We complement the discussion with a brief overview of recent advancements in diffusion models and graph neural networks.

Reassembly Tasks. Reassembly tasks captivate the attention of the research community as a benchmark for investigating the effectiveness of solutions that employ a reasoning process in the spatial domain. Here, we present the relevant works for the most common reassembly task in 2D and 3D: jigsaw puzzles and fracture object reassembly.

2D jigsaw puzzles. Puzzles are used to investigate the intricacies of image ordering with inherent combinatorial complexity [5]. Among the most successful strategies are those rooted in optimization and greedy approaches that rely on hand-crafted features [15, 22, 52]. More recently, there has been a shift towards employing learning-based methods to solve puzzles with only shifted pieces [18, 28, 34, 44]. These approaches demonstrate greater resilience when handling inputs with distortions, though they perform worse compared to optimization methods in standard scenarios. Moreover, these methods do not handle rotated pieces.

3D fractured objects. Fractured object reassembly in 3D is extensively explored in the literature [4, 14] and has applications in numerous fields, such as fresco reconstruction [2], and furniture assembly [27]. A recent effort in solving the problem was introducing the *Breaking Bad* dataset [37]. In *Breaking Bad*, the challenge involves reconstructing a broken object from multiple fragments. Those fragments are not labeled with any semantic information, as in many real-world applications [2]. Previous research efforts focus on predicting 6-degree-of-freedom poses for input parts (such as chair backs, legs, and bars) [54] and assembling 3D shapes from images of the complete object [29]. These prior investigations heavily lean on the semantic details of object

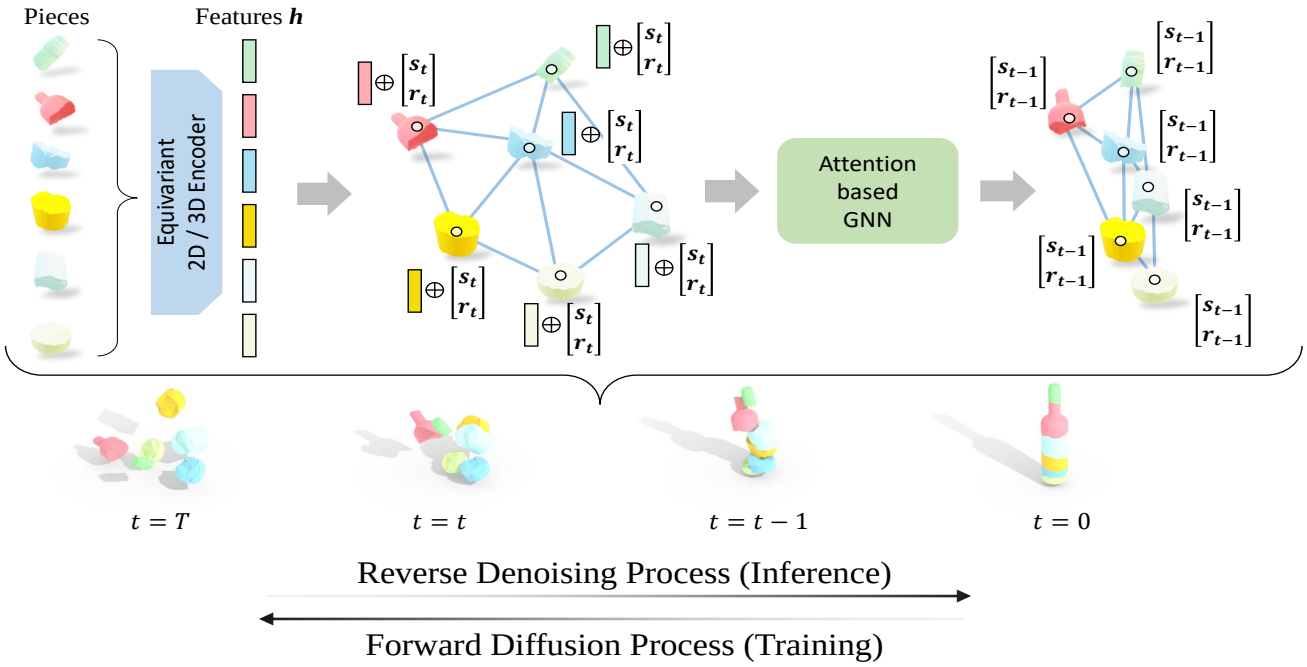

Figure 2. Framework of our proposed DiffAssemble for reassembly tasks, here is shown for the 3D task. Following the Diffusion Probabilistic Models formulations, we model a Markov chain where we inject noise into the pieces' position and orientation. At timestep $t = 0$, the pieces are in their correct position, and at timestep $t = T$, they are in a random position with random orientation. At each timestep t, our attention-based GNN takes as input a graph where each node contains an equivariant feature that describes a particular piece and its position and orientation. The network then predicts a less noisy version of the piece's position and orientation.

parts, overlooking essential geometric cues. Neural Shape Matching (NSM) [4] addressed the two-part mating problem by emphasizing shape geometries over semantic information. *SE(3)-Equiv* [50] tackles the problem with specific design choices that go beyond object reassembly, e.g., adversarial and reconstruction losses.

Unlike the previous works, which focus on just one aspect of the problem, we propose a unified solution for reassembly tasks. Furthermore, to the best of our knowledge, we are the first to propose a learning-based approach for puzzles with translated and rotated pieces.

Diffusion Probabilistic Models. DPMs are generative models that have shown remarkable results in recent years. These models approach the generation process through a bidirectional iterative chain. In one direction, they transform data into a Gaussian distribution by incrementally adding Gaussian noise. They are then trained to reverse this process, generating new samples from the initial distribution starting from random noise [19]. DPMs demonstrate remarkable versatility across a range of tasks applications, including image synthesis [10], semantic segmentation [1], and generation [31]. Their applicability extends to spatial data processing in both 2D and 3D contexts, where they have been effectively employed in object detection [3], scene generation [21], and 3D protein modeling [51].

Our work proposes the use of Diffusion Models for reassembly tasks. We introduce a unified model capable of operating effectively in both 2D and 3D space, aiming to retrieve the original position and orientation of the constituent pieces accurately.

Graph Neural Networks. Graph Neural Networks (GNNs) underwent significant advancements in recent years with new models like GCN [24], GraphSage [30], and SigMaNet [12]. These advancements have continued with new architectures [47, 55] that incorporate attention mechanisms that weigh the importance of nodes during message passing. The use of Graphs and GNNs has seen widespread adoption for spatial applications, as they are able to describe an arbitrary number of elements and their relation to each other. Common applications include scene graph generation [36], 3D scene generation [9], object localisation [17], relative pose estimation [43], and robot navigation [35]. Scalability is a common issue affecting GNNs and Exphormer [41] represents a significant stride in scalable graph transformer architectures, utilizing a sparse attention mechanism that leverages virtual global nodes and expander graphs [8].

We represent the puzzle as a graph and process it by using an Attention-based GNN [39]. We adopt the Exphormer [41] to enhance DiffAssemble's capability in handling the computational demands of reassembly tasks.

3. Our Method

We reassemble a set of elements by predicting the translation and rotation, i.e. the pose of a piece, with the objective to arrange the elements in a coherent structure, such as a non-broken 3D object or a solved 2D puzzle. Figure 2 summarizes our approach. We represent the set of pieces to reassemble as nodes in a complete graph, with each node having a modality-specific feature encoder. We adapt the DPM formulation [19] by introducing time-dependent noise into each element's translation and rotation. Noise injection resembles shuffling the set of elements, e.g., randomly distributing puzzles pieces or 3D object fragments in the 2D or 3D Euclidean space. During training, we process the noisy input graph via an Attention-based GNN to restore the initial translation and rotation of all elements. At inference, we initialize each element's starting positions and rotation from pure noise, and iteratively denoise the graph, reassembling the coherent structure in the process. Section 3.1 introduces our graph-based formulation. Section 3.2 discusses the input's feature representation. Section 3.3 presents our diffusion-based approach, and Section 3.4 defines our attention-based architecture and the sparsity mechanism.

3.1. Graph Formulation

We assign each of the M pieces m to a node v^m, defining a set of vertices $V = \{v^m\}_{m\in[1,\dots,M]}$. Since we do not want to introduce a priori relationship between the pieces, we connect all the nodes together. This defines a complete graph $G = (V, E)$, where E is the set of edges. We define the feature of each node v^m by concatenating the following vectors:

- **Features vector** $\mathbf{h}^m \in \mathbb{R}^d$, where d is the dimension of the feature generated by a equivariant encoder. DiffAssemble is agnostic to the adopted feature backbone.
- **Translation vector** $\mathbf{s}^m \in \mathbb{R}^n$, where n represents the dimensionality of the *continuous* Euclidean space in which the reassembly task is conducted.
- **Rotation matrix** $R^m \in SO(n)$, where $SO(n)$ is the Special Orthogonal Group in n dimensions. We also define a function $f_r(\mathbf{r}^m) = R^m$ that maps any vector rotation representation $\mathbf{r}^m$ to R^m.

The advantage of using this graph formulation lies in its ability to be flexible with respect to the cardinality of V. For this reason, DiffAssemble is able to work simultaneously with puzzles of various sizes rather than being limited to handling only one size at a time.

3.2. Feature Representation

A fundamental aspect of our architecture lies in its capability to operate with element features $\mathbf{h}^m$, which can be extracted by pre-trained encoders. Features play a central role in solving reassembly tasks, as they provide the network with inductive biases. This intuition is particularly relevant for complex tasks involving translation and rotation.

To extract the features, we first translate the piece so that its center lies on the origin and then use a rotation-equivariant encoder to map the visual and shape information into the latent space. Rotation-equivariant features undergo the same rotation that is applied in the original input space and are thus the best candidates to enable the neural network associating a specific rotation R^m (in the input space) to the features map $\mathbf{h}^m$. More details on group equivariance are given in the Supplementary Material.

3.3. Diffusion Models for Reassembly Tasks

We adopt Diffusion Probabilistic Models as defined in DDIM [42] to solve the reassembly tasks. We define a compact representation of the initial translation $\mathbf{s}_0^m$ and rotation $\mathbf{r}_0^m$ of piece m as a concatenated vector $\mathbf{x}_0^m = [\mathbf{s}_0^{m\top}, \mathbf{r}_0^{m\top}]^\top$.

At training time, we iteratively add noise sampled from a Gaussian Distribution $\mathcal{N}(0, I)$ to their poses (Forward Process). Following that, we train DiffAssemble to reverse this noising process (Reverse Process) and to obtain the initial poses $X_0 = \{\mathbf{x}_0^m\}_{m\in[1,\cdots,M]}$.

Forward Process. We define the forward process as a fixed Markov chain that adds noise following a Gaussian distribution to each input $\mathbf{x}_0^m$ to obtain a noisy version, $\mathbf{x}_t^m$, at timestep t. Following [19], we adopt the variance β_t according to a linear scheduler and define $q(\mathbf{x}_t^m|\mathbf{x}_0^m)$ as:

$$q(\mathbf{x}_t^m|\mathbf{x}_0^m) = \mathcal{N}(\mathbf{x}_t^m; \sqrt{\overline{\alpha}_t}\mathbf{x}_0^m, (1-\overline{\alpha}_t)\mathbf{I}), \tag{1}$$

where $\overline{\alpha}_t = \prod_{c=1}^{t}(1-\beta_c)$ and $\mathbf{I}$ is the indentity matrix.

Reverse Process. The reverse process iteratively retrieves the initial poses for the set of elements $\hat{X}_{t-1}$ given current (noisy) poses $X_t = \{\mathbf{x}_t^m\}_{m\in[1,\cdots,M]}$ and the features $H = \{\mathbf{h}^m\}_{m\in[1,\cdots,M]}$. $\hat{X}_{t-1}$ is computed as:

$$\hat{X}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(X_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(X_t, H, t)\right), \tag{2}$$

where $\alpha_t = 1-\beta_t$, and $\epsilon_\theta(X_t, H, t)$ is the estimated noise output by DiffAssemble that has to be removed from $\hat{X}_t$ at timestep t to recover $\hat{X}_{t-1}$.

Losses. Following a standard practice in Diffusion Models [19], we train DiffAssemble to predict $\hat{X}_0$ instead of $\hat{X}_{t-1}$. We introduce two loss functions to reconstruct the intial pose of each piece.

Translation Loss. This loss computes the average difference between the ground truth translation vectors and the

predicted ones $\hat{\mathbf{s}}_0^m$:

$$L_{tr} = \frac{1}{M}\sum_{m=1}^{M} \|\mathbf{s}_0^m - \hat{\mathbf{s}}_0^m\|_2^2,$$

where $\|\cdot\|_2^2$ is the squared L2 norm.

Rotation Loss. This loss measures the average difference between the ground truth rotation matrices and the predicted ones $f_r(\hat{\mathbf{r}}_0^m)$:

$$L_{rt} = \frac{1}{M}\sum_{m=1}^{M} \|f_r(\mathbf{r}_0^m)^\top f_r(\hat{\mathbf{r}}_0^m) - \mathbf{I}\|_2^2.$$

3.4. Architecture

We use an Attention-based GNN with $L-$layers of Unified Message Passing (UniMP) [40]. UniMP implements a multi-head attention mechanism over all nodes to scale the information gathered from neighboring nodes during message passing. Multi-head attention is well-suited for graph contexts where we lack prior knowledge of node relationships, i.e., we cannot define an adjacency matrix A.

However, one of the main constraints associated with these attention-based architectures [47, 55] lies in the inherent definition of a complete graph. This constraint poses a severe limitation for scaling on large graphs, i.e., dealing with a large number of elements. To address this constraint, we employ exphormer [41], which relies on the expander graph [20] and virtual nodes to reduce memory requirements by cleverly pruning edges in the graph. In Section 4.3, we show how we use exphormer to scale to large graphs.

4. Experimental Evaluation

DiffAssemble tackles 3D objects reassembly and 2D jigsaw puzzles as two possible instantiations of a *reassembly task*. We first validate DiffAssemble on 3D object reassembly (Section 4.1), showing through quantitive and quality results the benefits of our approach. Section 4.2 discusses the performance of our approach on 2D jigsaw puzzles in the standard scenario and when dealing with missing pieces. We carry out an ablation study of DiffAssemble's design choices in Section 4.1 and in the Supplementary Material. Finally, we tackle DiffAssemble's limitation on large puzzles in Section 4.3, demonstrating that DiffAssemble efficiently reassemble up to 900 elements thanks to the sparsity mechanism.

Throughout this section, tables report the best results in **boldface** and the second-best underlined.

4.1. 3D Object Reassembly

First, we explore the application of DiffAssemble to the task of reassembling objects in 3D.

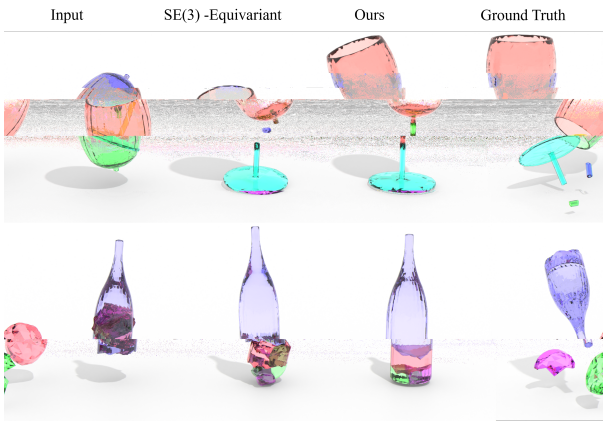

Figure 3. Qualitative results on Breaking Bad, showing the reassembly results for a broken wine glass and a wine bottle. We compare the results against SE(3)-Equiv [50], which is the current SOTA method. All results are in the same reference frame, shifted horizontally so they do not overlap. We show the results with glass materials to better show overlapping pieces.

Dataset and Evaluation Setting. We test our methods on 3D object reassembly on Breaking Bad (BB) [37]. It is composed of 3D meshes for 20 classes of everyday objects, such as bottles, plates, glasses. For each of these objects, there are multiple variants, where the object is broken into multiple parts by simulating fractures in the geometry. The dataset provides objects split into 2 to 100 pieces. As proposed in BB, we train and test using objects composed of 2 to 20 parts. All the objects' pieces are translated to the origin and randomly rotated. For each piece, we need to provide a pose that returns the fragment to its correct location in the object's canonical pose. Following the evaluation pipeline in [37], we report the metrics in terms of Root Mean Squared Error in rotation RMSE (R), Root Mean Square Error in translation RMSE (T), and Part Accuracy (PA), which measures the percentage of parts whose Chamfer Distance to ground-truth is less than 0.01 [54]. We compare with the three baselines proposed in BB: Global, DGL, LSTM. In addition, we compare with SE(3)-Equiv. [50], the current state of the art on BB, which integrates both equivariant and invariant features. Regarding our approach, we use the base model along with two variations: one without the diffusion process, predicting the translation and rotation of the pieces in a single step, and another version that omits the use of an equivariant encoder.

Implementation Details. Each fragment m of a 3D object is a point-cloud of $1,000$ points. For the 3D shape reassembly, we use VN-DGCNN [50] as our feature extractor. This backbone takes as input the point cloud of each piece and returns both an equivariant and invariant representation. We only consider the equivariant features to create the element features $\mathbf{h}^m$. For this task, we parameterize the rotation in 3D as a unit quaternion, $q^m = q_0^m + \mathbf{i}q_1^m + \mathbf{j}q_2^m + \mathbf{k}q_3^m$, where

METHOD	RMSE (R) # degree	RMSE (T) # $\times 10^{-2}$	PA " %
Global [37]	81.6	15.2	17.5
DGL [37]	81.4	<u>14.9</u>	<u>25.4</u>
LSTM [37]	87.4	15.8	11.3
NSM [4]†	83.3	15.3	10.6
SE(3)-Equiv [50]	<u>77.9</u>	16.7	8.1
DiffAssemble - No Diffusion Process	83.6	17.1	3.1
DiffAssemble - No Equivariant Enc.	81.7	17.0	18.3
DiffAssemble	**73.3**	**14.8**	**27.5**

Table 1. Quantitative results of four learning-based shape reassembly baselines and DiffAssemble on the `everyday` object subset. †Modified version, suggested in [50], capable of handling multi-part assembly.

STAGE	CHANGES	RMSE (R) # degree	RMSE (T) # $\times 10^{-2}$	PA " %
Representation	Non-Equivariant Enc.	81.68	17.04	18.32
	Invariant Enc.	77.06	18.09	14.27
Diff. Design	6 degree-of-freedom rotation	75.60	18.80	18.50
	w/o Chamfer Distance loss	**72.75**	**14.78**	<u>24.10</u>
	w/ Chamfer Distance loss	<u>73.34</u>	<u>14.82</u>	**27.48**
	No Diff. process	83.60	17.12	3.10
GNN	Standard GCN [24]	74.56	15.79	21.33

Table 2. Ablation for 3D object reassembly on the `everyday` object subset.

q_0^m, q_1^m, q_2^m and q_3^m are real numbers, $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$ are mutually orthogonal basis vectors. Thus, we define the vector $\mathbf{r}^m = [q_0^m, q_1^m, q_2^m, q_3^m]^\top$. Since we parameterize rotations as unit quaternions, i.e., $\boldsymbol{j}\mathbf{q}^m\boldsymbol{j} = 1$, the direct application of the forward process of the diffusion steps is not feasible as it may generate rotation values outside the $SO(3)$ manifold [25]. Following [25], we address this limitation by leveraging the diffusion processes on the Lie group $SO(3)$ (more details are available in the Supplementary Material). As proposed in [37], we also test the effect of an additional Chamfer Distance Loss term.

Results. We report in Table 1 the results of the comparison on BB. Among the baselines, SE(3)-Equiv, which is the current SOTA, performs best in terms of RMSE(R), and DGL performs best in terms of RMSE(T) and PA. These baselines trade accuracy in rotation with accuracy in translation, with SE(3)-Equiv performing well in rotation and worst in translation and DGL performing well in translation and badly in rotation. Contrarily, DiffAssemble outperforms the baselines on all metrics: rotation, translation, and part accuracy, showing the effectiveness of our approach.

When comparing the two variants of our approach, we see the importance of using both an equivariant feature representation and the diffusion process. Notably, we observe significantly worse results when one of these elements is missing: RMSE(R) drops by 10 points, RMSE(T) drops by 3 points, and part accuracy drops from 27% to 3%.

In the following paragraph, we report the results with other variants of our approach. Figure 3 reports a qualitative comparison between DiffAssemble and the SE(3)-Equiv when reassembling a wine glass and a bottle, both fragmented in four pieces. We observe that SE(3)-Equivariant struggles in dealing with both large and small pieces, and shifts all pieces to the middle point. Contrarily, DiffAssemble was able to handle big pieces well but struggled with small pieces like the fragments of the glass stem.

Ablation. Table 2 reports results assessing *i)* the importance of the feature representation, *ii)* the impact of the diffusion design, and *iii)* the benefit of using attention.

We notice that employing invariant and non-equivariant features leads to worse performance. This result highlights the importance of providing the network with inductive biases, specifically rotation-equivariant feature, to solve this task. In the Diffusion Design section of the table, we compare our implemented forward process for handling rotation in $SO(3)$ with a direct application of Gaussian Noise to the 6D representation (6DOF) [56]. This straightforward approach negatively impacts the model's performance across all metrics. Following [37], we investigate the use of the Chamfer Distance (CD) loss alongside our general losses. We see that by using the CD loss, we improve in Part Accuracy but perform worse in both RMSE for translation and rotation. Nevertheless, the loss in performance is minor, and with or without Chamfer Distance loss, aside from the part

Method		Dataset							
		PuzzleCelebA				PuzzleWikiArts			
		6x6	8x8	10x10	12x12	6x6	8x8	10x10	12x12
Optimization Based	Gallagher [15]	80.21	55.18	71.19	69.81	71.88	61.63	54.15	44.68
	Yu *et al*. [52]	98.63	94.65	98.33	93.33	**94.62**	**92.95**	**90.99**	**89.88**
	Huroyan *et al*. [22]	98.47	**97.45**	98.65	97.08	92.69	91.37	89.74	88.28
Learning Based	**DiffAssemble** - No Diff.	99.43	79.84	99.05	91.28	73.07	54.70	22.68	18.27
	DiffAssemble - No Equiv.	96.12	71.62	91.98	64.15	25.31	14.63	8.19	4.96
	DiffAssemble	**99.51**	87.66	**99.30**	**97.76**	90.65	72.79	63.33	53.08

Table 3. Results for Jigsaw puzzle solving on *PuzzleCelebA* and *PuzzleWikiArts*

Method	Dataset			
	CelebA		WikiArts	
	6x6	12x12	6x6	12x12
Gallagher [15]	33.28 *(-46.93)*	19.18 *(-50.63)*	32.19 *(-39.69)*	24.12 *(-20.56)*
Yu [22]	33.45 *(-66.85)*	21.78 *(-72.84)*	32.53 *(-62.09)*	24.65 *(-65.23)*
Huroyan [52]	18.18 *(-80.29)*	0.09 *(-88.45)*	17.14 *(-75.55)*	0.08 *(-80.28)*
DiffAssemble	**96.92** *(-2.59)*	**76.49** *(-32.81)*	**51.21** *(-39.44)*	**27.09** *(-25.99)*

Table 4. Results for Jigsaw puzzle solving with 30% missing pieces on *PuzzleCelebA* and *PuzzleWikiArts*. The percentage variance of the model relative to the result presented in Table 3 is reported within square brackets.

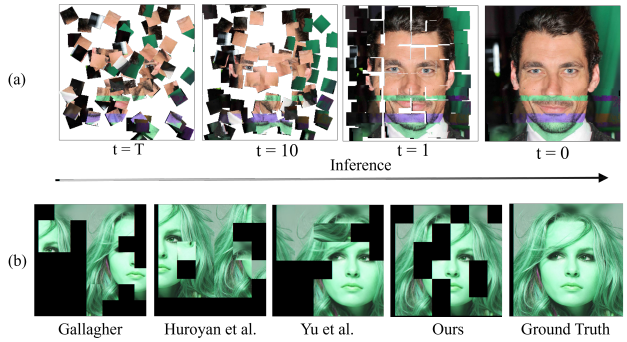

Figure 4. **(a)** Patch alignment in inference from $t = T$ to $t = 0$. **(b)** Qualitative comparison with 30% missing pieces.

accuracy, our method performs best.

Finally, we investigate the impact of the attention mechanism on information propagation. For this purpose, we define the adjacency matrix $A = \mathbb{1} \in \mathbb{R}^{M \times M}$ as an all-ones matrix and, instead of UniMP, we use the Graph Convolutional Network (GCN) [24]. DiffAssemble with UniMP consistently outperforms DiffAssemble with GCN, highlighting the benefit of adopting an attention mechanism.

4.2. 2D Jigsaw Puzzle with Rotated Pieces

We adopt DiffAssemble to reassemble visual jigsaw puzzles with translated and rotated pieces. In this task, we need to reassemble an image by translating and rotating a collection of image patches. The patches are regularly shaped, non-overlapping, and initialized with a random orientation.

Dataset and Evaluation Setting. Following [44], we test our approach on two datasets of puzzles: PuzzleCelebA [26] and PuzzleWikiArt [45]. PuzzleCelebA is a dataset of celebrities' faces, while PuzzleWikiArt contains paintings from various artists in many styles. We compare with three optimization-based methods for visual puzzle-solving: *i)* Gallagher [15], *ii)* Yu *et al*. [52], and *iii)* Huroyan *et al*. [22].

We report the results using various numbers of patches, from 36 (6 × 6) to 144 (12 × 12). We present the outcomes using the *direct comparison metric* [5], where a piece is successfully placed if it is both correctly positioned and rotated. While our solution operates in the continuous domain, the evaluation is conducted in the discrete one of [15]. To do so, we first apply to each piece the predicted (continuous) translation and rotation; then, we discretize its pose by snapping the piece to the closest cell in a $n \times n$ squared lattice, $n = \sqrt{M}$, and its rotation to the nearest $\pi/2$ angle.

Implementation Details. Each piece has to be arranged into a two-dimensional continuous space with boundaries $[-1, 1]$. Following [15], the possible initial rotation of each patch is represented by the set $\{0, \pi/2, \pi, 3\pi/2\}$, which are elements of the cyclic group C_4 [15]. We parameterized the rotation in 2D as: $\mathbf{r}^m = [\cos(\theta^m), \sin(\theta^m)]^\top$ [56] and we optimize θ^m in the continuous domain. For the feature extractor, we use a version of ResNet18 that is equivariant [6] to the cyclic group C_4.

Results. Table 3 reports results for the visual puzzle reassembly task, with rotated and translated pieces. DiffAssemble achieves SOTA results in CelebA, improving over the optimization-based method. In Wikiart, the optimization-based approaches [22, 52] outperform DiffAssemble. An explanation for this gap is that our method relies not only on pure visual appearances but also on the semantic content of the images. CelebA has a very strong semantic structure, containing images of faces in similar

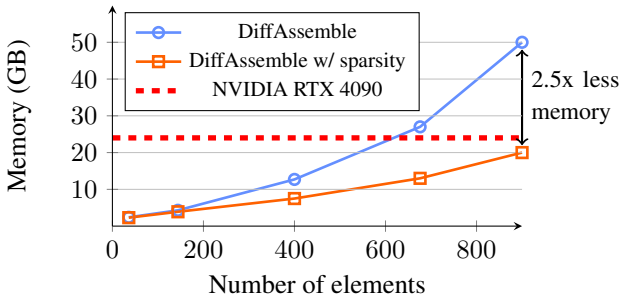

Figure 5. GPU memory consumption by total size.

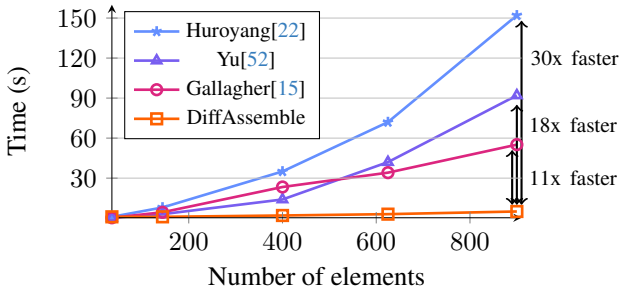

Figure 6. Time requirement to solve one puzzle for our approach and the optimization methods.

poses. In contrast, Wikiart has very diverse images with no predefined structure. Optimization-based approaches directly match visual content along the borders. This design makes them a strong baseline when all patches are provided, at a cost of time efficiency, as shown in Figure 6. Section 4.3 further discusses efficiency as we scale up to larger puzzles.

Relying on semantics also makes DiffAssemble's robust when some of the pieces are missing. Testing robustness to missing pieces is common when solving jigsaw puzzles [15, 44], as it reflects real-world application, e.g., fresco reconstruction [2]. We evaluate DiffAssemble and all the baselines when 30% of the pieces are randomly removed and report results in Table 4. DiffAssemble outperforms Huroyan *et al*. [52], the second-best model, in both CelebA and Wikiart. Optimization-based methods experience a significant decrease in accuracy on both 6×6 and 12×12 puzzles, while DiffAssemble retains high performances even in this challenging setting. Figure 4 shows DiffAssemble solving a CelebA puzzle from randomly shuffled pieces, along with a comparison with all the baselines when 30% of the pieces are missing. In the Supplementary Material, we present an ablation on the design choices for 2D Jigsaw puzzle, analogously to the above-mentioned ablation study for 3D object reassembly.

4.3. Scaling to Larger Graphs

We investigate DiffAssemble with Exphander [41] for higher-dimensional puzzles. We explore the effectiveness of scaling our method with PuzzleCelebA puzzles up to 900 pieces (30×30 puzzles). DiffAssemble with Expander prunes 80% of the edges from the complete graph during training and introduces 8 virtual nodes to ensure global connectivity. Figure 5 shows the memory requirements for DiffAssemble with and without sparsity, executed on standard consumer-grade hardware (NVIDIA GeForce RTX 4090 with 24GB). When the graph has 900 elements, our method with sparsity halves the memory consumption without compromising accuracy.

Although our method requires much memory, it is significantly faster than memory-intensive optimization methods. We compare DiffAssemble with the three optimization-based approaches. Figure 6 reports the time required for the four methods to solve a puzzle based on size. The time required by optimization-based approaches increases exponentially with the number of elements and, consequently, with the number of matches. On the other hand, DiffAssemble reassembles up to 900 elements without scaling in time requirement, e.g., it solves 30×30 puzzles in 5s with 95% accuracy. This represents a significant improvement over Gallagher, the faster optimization-based solution, which has a run-time of 55s with an accuracy of 58%.

5. Conclusion

In this work, we introduced **DiffAssemble**, a general framework for tackling reassembly tasks through graph representations and a diffusion model formulation. By framing reassembly as a denoising task, we leverage an Attention-based Graph Neural Network to iteratively refine the pose of each piece through a diffusion process.

Our experimental evaluation showcases the effectiveness of DiffAssemble, spanning 3D object reassembly and 2D puzzles with translated and rotated pieces. The results demonstrate SOTA performance in most 2D and 3D scenarios, revealing a common ground between these seemingly disparate tasks. Notably, in the 2D domain, DiffAssemble exhibits robustness to missing pieces and achieves remarkable efficiency compared to optimization-based methods. In the 3D, our solution obtains SOTA results, maintaining accuracy in translation and rotation, unlike previous solutions.

Limitations and Future Research. One of the main limitations of DiffAssemble is its high memory usage, even by introducing the sparsity mechanism based on the expander graph. Future efforts will focus on mitigating the memory demands and exploring further reassembling scenarios while dealing with data from real-world scans.

Acknowledgments This work is part of the RePAIR project that has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 964854 and is supported by the project Future Artificial Intelligence Research (FAIR) – PNRR MUR Cod. PE0000013 - CUP: E63C22001940006.

References

[1] Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *International Conference on Learning Representations*, 2021. 3

[2] Benedict J Brown, Corey Toler-Franklin, Diego Nehab, Michael Burns, David Dobkin, Andreas Vlachopoulos, Christos Doumas, Szymon Rusinkiewicz, and Tim Weyrich. A system for high-volume acquisition and matching of fresco fragments: Reassembling theran wall paintings. *ACM transactions on graphics (TOG)*, 27(3):1–9, 2008. 1, 2, 8

[3] Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19830–19843, 2023. 3

[4] Yun-Chun Chen, Haoda Li, Dylan Turpin, Alec Jacobson, and Animesh Garg. Neural shape mating: Self-supervised object assembly with adversarial shape priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12724–12733, 2022. 2, 3, 6

[5] T. S. Cho, S. Avidan, and W. T. Freeman. A probabilistic image jigsaw puzzle solver. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 183–190, 2010. 2, 7

[6] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016. 7

[7] Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. *International Conference on Learning Representations (ICLR)*, 2023. 1

[8] Andreea Deac, Marc Lackenby, and Petar Veličković. Expander graph propagation. In *Learning on Graphs Conference*, pages 38–1. PMLR, 2022. 3

[9] Helisa Dhamo, Fabian Manhardt, Nassir Navab, and Federico Tombari. Graph-to-3d: End-to-end generation and manipulation of 3d scenes using scene graphs. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 3

[10] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021. 3

[11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011. 1

[12] Stefano Fiorini, Stefano Coniglio, Michele Ciavotta, and Enza Messina. Sigmanet: One laplacian to rule them all. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7568–7576, 2023. 3

[13] Herbert Freeman and L Garder. Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition. *IEEE Transactions on Electronic Computers*, (2):118–127, 1964. 1

[14] Thomas Funkhouser, Hijung Shin, Corey Toler-Franklin, Antonio García Castañeda, Benedict Brown, David Dobkin, Szymon Rusinkiewicz, and Tim Weyrich. Learning how to match fresco fragments. *Journal on Computing and Cultural Heritage (JOCCH)*, 4(2):1–13, 2011. 2

[15] Andrew C Gallagher. Jigsaw puzzles with pieces of unknown orientation. In *2012 IEEE Conference on computer vision and pattern recognition*, pages 382–389. IEEE, 2012. 1, 2, 7, 8

[16] Howard E Gardner. *Frames of mind: The theory of multiple intelligences*. Basic books, 2011. 1

[17] Francesco Giuliari, Geri Skenderi, Marco Cristani, Yiming Wang, and Alessio Del Bue. Spatial commonsense graph for object localisation in partial scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19518–19527, 2022. 3

[18] Francesco Giuliari, Gianluca Scarpellini, Stuart James, Yiming Wang, and Alessio Del Bue. Positional diffusion: Ordering unordered sets with diffusion probabilistic models. *arXiv preprint arXiv:2303.11120*, 2023. 2

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 3, 4

[20] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006. 5

[21] Siyuan Huang, Zan Wang, Puhao Li, Baoxiong Jia, Tengyu Liu, Yixin Zhu, Wei Liang, and Song-Chun Zhu. Diffusion-based generation, optimization, and planning in 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16750–16761, 2023. 3

[22] Vahan Huroyan, Gilad Lerman, and Hau-Tieng Wu. Solving jigsaw puzzles by the graph connection laplacian. *SIAM Journal on Imaging Sciences*, 13(4):1717–1753, 2020. 1, 2, 7, 8

[23] Benjamin Jones, Dalton Hildreth, Duowen Chen, Ilya Baran, Vladimir G Kim, and Adriana Schulz. Automate: A dataset and learning approach for automatic mating of cad assemblies. *ACM Transactions on Graphics (TOG)*, 40(6):1–18, 2021. 3

[24] Thomas. N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th*

International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings, 2017. 3, 6, 7, 2

[25] Adam Leach, Sebastian M Schmon, Matteo T Degiacomi, and Chris G Willcocks. Denoising diffusion probabilistic models on so (3) for rotational alignment. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022. 6, 1

[26] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 7, 3

[27] Youngwoon Lee, Edward S Hu, and Joseph J Lim. Ikea furniture assembly environment for long-horizon complex manipulation tasks. In *2021 ieee international conference on robotics and automation (icra)*, pages 6343–6349. IEEE, 2021. 2

[28] Ru Li, Shuaicheng Liu, Guangfu Wang, Guanghui Liu, and Bing Zeng. Jigsawgan: Auxiliary learning for solving jigsaw puzzles with generative adversarial networks. *IEEE Transactions on Image Processing*, 31: 513–524, 2021. 2

[29] Yichen Li, Kaichun Mo, Lin Shao, Minhyuk Sung, and Leonidas Guibas. Learning 3d part assembly from a single image. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 664–682. Springer, 2020. 2

[30] Jielun Liu, Ghim Ping Ong, and Xiqun Chen. Graphsage-based traffic speed forecasting for segment network with sparse data. *IEEE Transactions on Intelligent Transportation Systems*, 23(3):1755–1766, 2020. 3

[31] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021. 3

[32] William Marande and Gertraud Burger. Mitochondrial dna as a genomic jigsaw puzzle. *Science*, 318(5849): 415–415, 2007. 1

[33] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2019. 3

[34] Marie-Morgane Paumard, David Picard, and Hedi Tabia. Deepzzle: Solving visual jigsaw puzzles with deep learning and shortest path optimization. *IEEE Transactions on Image Processing*, 29:3569–3581, 2020. 2

[35] Zachary Ravichandran, Lisa Peng, Nathan Hughes, J. Daniel Griffith, and Luca Carlone. Hierarchical representations and explicit memory: Learning effective navigation policies on 3d scene graphs using graph neural networks. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 9272–9279, 2022. 3

[36] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone. 3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans. In *Robotics: Science and Systems (RSS)*, 2020. 3

[37] Silvia Sellán, Yun-Chun Chen, Ziyi Wu, Animesh Garg, and Alec Jacobson. Breaking bad: A dataset for geometric fracture and reassembly. *Advances in Neural Information Processing Systems*, 35:38885–38898, 2022. 2, 5, 6, 3

[38] Jean-Pierre Serre et al. *Linear representations of finite groups*. Springer, 1977. 1

[39] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*, 2020. 3

[40] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2021. 2, 5

[41] Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J Sutherland, and Ali Kemal Sinop. Exphormer: Sparse transformers for graphs. *arXiv preprint arXiv:2303.06147*, 2023. 2, 3, 5, 8

[42] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 4

[43] Matteo Taiana, Matteo Toso, Stuart James, and Alessio Del Bue. Posernet: Refining relative camera poses exploiting object detections. In *European Conference on Computer Vision*, pages 247–263. Springer, 2022. 3

[44] Davide Talon, Alessio Del Bue, and Stuart James. Ganzzle: Reframing jigsaw puzzle solving as a retrieval task using a generative mental image. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 4083–4087. IEEE, 2022. 1, 2, 7, 8

[45] Wei Ren Tan, Chee Seng Chan, Hernan E Aguirre, and Kiyoshi Tanaka. Improved artgan for conditional synthesis of natural image and artwork. *IEEE Transactions on Image Processing*, 28(1):394–409, 2018. 7

[46] Wei Ren Tan, Chee Seng Chan, Hernan Aguirre, and Kiyoshi Tanaka. Improved artgan for conditional synthesis of natural image and artwork. *IEEE Transactions on Image Processing*, 28(1):394–409, 2019. 3

[47] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio.

Graph attention networks. In *International Conference on Learning Representations*, 2018. 3, 5

[48] Ariana M Villegas-Suarez, Cristian Lopez, and Ivan Sipiran. Matchmakernet: Enabling fragment matching for cultural heritage analysis. *Preprint*. Accessed: No DOI available at the time of retrieval. 1

[49] Karl DD Willis, Pradeep Kumar Jayaraman, Hang Chu, Yunsheng Tian, Yifei Li, Daniele Grandi, Aditya Sanghi, Linh Tran, Joseph G Lambourne, Armando Solar-Lezama, et al. Joinable: Learning bottom-up assembly of parametric cad joints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15849–15860, 2022. 3

[50] Ruihai Wu, Chenrui Tie, Yushi Du, Yan Zhao, and Hao Dong. Leveraging se(3) equivariance for learning 3d geometric shape assembly. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14311–14320, 2023. 3, 5, 6

[51] Jason Yim, Brian L Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. Se (3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023. 3

[52] Rui Yu, Chris Russell, and Lourdes Agapito. Solving jigsaw puzzles with linear programming. *arXiv preprint arXiv:1511.04472*, 2015. 1, 2, 7, 8

[53] E Yurteri, Sabri Derin, Fatma Polat, Murat Canpolat, Zekiye Barman, C Polat, Emel Kaya, and Selim Güllü. Obstacle maps. *New Era Journal*, 1(1):1–15, 2022. 1

[54] Guanqi Zhan, Qingnan Fan, Kaichun Mo, Lin Shao, Baoquan Chen, Leonidas J Guibas, Hao Dong, et al. Generative 3d part assembly via dynamic graph learning. *Advances in Neural Information Processing Systems*, 33:6315–6326, 2020. 2, 5

[55] Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. Graph-bert: Only attention is needed for learning graph representations. *arXiv preprint arXiv:2001.05140*, 2020. 3, 5

[56] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019. 6, 7

DiffAssemble: A Unified Graph-Diffusion Model for 2D and 3D Reassembly

Supplementary Material

A. Experiment Details

Hardware. The experiments were conducted on 2 different machines: four NVIDIA Tesla V100 16GB, 380 GB RAM, and 2x Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz Sky Lake CPU, and one NVIDIA RTX 4090 GPU, 64 GB RAM, and 12th Gen Intel(R) Core(TM) i9-12900KF CPU @ 3.20GHz CPU.

Model Settings. We train DiffAssemble with a learning rate of 10^{-4} and Adagrad as the optimization algorithm [11]. During our training process, we set a maximum of 1000 epochs, but we stop the training earlier to prevent unnecessary iterations when the loss no longer decreases.

B. Equivariant Feature Representation

As we presented in Section 3.2, one of key point of our proposal lies in its ability to work with element features $\mathbf{h}^m$, which can be extracted by any pre-trained encoders. In particular, we discover the importance to extract rotation-equivariant features.

A function ϕ is equivariant to the action of a group G if $\phi(S_g(\cdot)) = S_g^{'}(\phi(\cdot))$ for all $g \in G$, where S_g and $S_g^{'}$ are linear representations related to the group element g [38]. This means that applying ϕ to the codomain of $S_g(\cdot)$ is equivalent to applying $S_g^{'} \in G$ to the codomain of ϕ. In this work, the transformation S_g and $S_g^{'}$ are rotations. As a result, the equivariant function $\phi(\cdot)$, i.e. the backbone, ensures the consistency of the rotational effect irrespective of whether it is applied before or after the function. Consequently, DiffAssemble associates a specific rotation $\mathbf{r}^m$ (in the input space) to the features vector $\mathbf{h}^m$.

C. Diffusion process and Rotation in 3D

We provide a more detailed description of how we introduce Gaussian Noise with 3D rotations. Following [25], we use a specific procedure to scale the rotation matrices $f_r(\mathbf{r}_t^m)$ by *i)* converting the rotation matrix to values in the Lie algebra $\mathfrak{so}(3)$, *ii)* multiplying them element-wise with t-dependent scalars, and *iii)* converting back to a rotation matrix through matrix exponentiation. Analogous to an addition in Euclidean space, the composition of rotations is done through matrix multiplication in $SO(3)$ as:

$$\lambda(\gamma_t, \mathbf{r}_t^m) = \exp(\gamma_t \log(f_r(\mathbf{r}_t^m))),$$

where $\lambda(\dots)$ is the geodesic distance flow from $\mathbf{I}$, the identity matrix, to $\mathbf{r}_t^m$ by an amount γ_t.

In particular, for the Forward Process, we rewrite Equation (1) to inject noise into $\mathbf{r}_0^m$:

$$q(\mathbf{r}_t^m | \mathbf{r}_0^m) = IG_{\text{SO(3)}}(\lambda(\sqrt{\overline{\alpha}_t}, \mathbf{r}_0^m), (1 - \overline{\alpha}_t)),$$

where $IG_{\text{SO(3)}}$ is the isotropic Gaussian distribution (IG) that is compatible with $SO(3)$ rotation directly. The IG distribution is parameterized in an axis-angle form by sampling uniformly an axis and rotation angle $\omega \in [0, \pi]$ as:

$$f(\omega) = \frac{1 - \cos\omega}{\pi} \sum_{l=0}^{\infty} (2l+1) e^{-l(l+1)\epsilon^2} \frac{\sin((l+0.5)\omega}{\sin(\omega/2)}.$$

For the Reverse Process, letting $R_t = \{\mathbf{r}_t^m\}_{m \in [1,\cdots,M]}$ and $H = \{\mathbf{h}^m\}_{m \in [1,\cdots,M]}$, we rewrite Equation (2) as follows:

$$\hat{R}_{t-1} = \lambda\left(\frac{\sqrt{\overline{\alpha}_{t-1}}}{\alpha_t}, R_t\right) \lambda\left(\frac{1 - \overline{\alpha}_{t-1}}{\sqrt{1 - \overline{\alpha}_t}}, \epsilon_\theta^{\text{rot}}(R_t, t, H)\right)^T,$$

where $\epsilon_\theta^{\text{rot}}(R_t, t, H)$ is the estimated noise that has to be removed from R_t to recover $\hat{R}_{t-1}$.

D. Additional Ablations

Missing Fragments in 3D Objects Reassembly We assess the performance of DiffAssemble and the baselines in scenarios involving missing 3D pieces. We consider a setting where each object is composed of 10 to 20 parts. We test the methods in four different scenarios: *i)* without missing pieces, *ii)* 10% of missing pieces, *iii)* 20% missing pieces, and *iv)* 30% of missing pieces. We do not retrain the models with missing pieces, but instead, we use the same method and weights as in the main paper experiment described in Section 4.1. To account for potential variations in fracture sizes within each object, we report the experiment five times using different seeds. This methodology helps alleviate potential biases introduced by excluding fractures with differing levels of complexity. Mean and standard deviation for each metric provide an indication of the overall behavior of the compared methods.

Table 2 reports the results, demonstrating that in all four scenarios, DiffAssemble outperforms the baseline in 2 out of 3 metrics. There is a decrease in performance when we increase the number of missing pieces, even if this reduction is minimal.

2D Jigsaw Puzzle. Table 6 reports further ablation results for the puzzle setting, which we could not include in the main paper due to space constraints.

Missing	0%			10%		
Method	RMSE (R) # degree	RMSE (T) # $\times 10^{-2}$	PA " %	RMSE (R) # degree	RMSE (T) # $\times 10^{-2}$	PA " %
Global	83.00	18.74	7.02	83.86	<u>18.76</u>	6.78
DGL	84.56	**18.26**	<u>9.72</u>	84.74	18.98	<u>8.42</u>
LSTM	88.26	19.64	4.78	88.40	19.74	4.96
SE(3)-Equiv	<u>81.82</u>	<u>18.50</u>	6.74	<u>82.96</u>	**18.54**	6.58
DiffAssemble	**80.13**	19.02	**11.61**	**80.32**	19.32	**11.20**

Missing	20%			30%		
Method	RMSE (R) # degree	RMSE (T) # $\times 10^{-2}$	PA " %	RMSE (R) # degree	RMSE (T) # $\times 10^{-2}$	PA " %
Global	84.20	<u>18.86</u>	6.66	84.76	**18.96**	<u>6.62</u>
DGL	85.01	19.80	<u>7.34</u>	85.64	20.68	6.56
LSTM	88.72	19.90	4.88	88.96	20.01	4.36
SE(3)-Equiv	<u>82.52</u>	**18.72**	6.54	<u>82.88</u>	<u>19.48</u>	6.51
DiffAssemble	**80.37**	19.52	**10.67**	**80.46**	19.84	**10.43**

Table 5. Results for DiffAssemble on BB's objects with 8-20 pieces when 0%/10%/20%/30% of the pieces are missing pieces. Our approach is robust even in the hardest scenario where 30% of the pieces are missing.

STAGE	CHANGES	PuzzleCelebA				PuzzleWikiArts			
		6x6	**8x8**	**10x10**	**12x12**	**6x6**	**8x8**	**10x10**	**12x12**
Representation	Non-Equivariant Enc.	96.12	71.62	91.98	64.15	25.31	14.63	8.19	4.96
	Invariant Enc.	22.97	20.01	16.87	13.63	7.64	4.64	2.79	1.66
Diff. Process	No Diff. process	99.43	79.84	99.05	91.28	73.07	54.70	22.68	18.27
GNN	Standard GCN [24]	85.03	54.35	71.19	45.56	30.12	22.07	10.77	1.08
DiffAssemble	Base Implementation (Tab. 3)	**99.51**	**84.94**	**99.30**	**97.76**	**90.65**	**72.79**	**63.33**	**53.08**

Table 6. We conduct an ablation study to evaluate the impact of each component of DiffAssemble for Jigsaw puzzle solving on *PuzzleCelebA* and *PuzzleWikiArts*. The base implementation corresponds to our proposed approach, as reported in Table 3 of the main paper.

We assess the benefit of employing rotation-equivariant features, instead of invariant and non-equivariant ones. These two last representations lead to worse performance in both datasets. In particular, these differences are more evident with the WikiArt dataset. DiffAssemble obtains an average improvement of 94.41% and 82.43% compared to DiffAssemble with invariant and non-equivariant features. This result highlights, one more time, the importance of employing rotation-equivariant features to solve reassembly tasks when rotation is involved.

We aimed at demonstrating that the adoption of the diffusion process is well-founded and effective. For this reasons, we experiment DiffAssemble wihtout the diffusion process. The results show that predicting the pose without the diffusion process, i.e., in 1 step, leads to worst performance, which serves as strong justification for the inclusion and use of the diffusion process in our approach.

Finally, we conducted an ablation for the *GNN architecture* adopted in DiffAssemble. Specifically, we assess the Graph Convolutional Network (GCN) [24] against UniMP. The goal is to investigate the impact of the attention mechanism on information propagation. For this purpose, we define the adjacency matrix $A \; 2 \; R^{M \times M}$ of the GCN as an all-ones matrix. Tables 6 reports the results of this comparison in the 2D and 3D scenarios, respectively. DiffAssemble with the use of UniMP consistently outperforms DiffAssemble with GCN, showing a remarkable improvement. These results highlight the importance of employing a mechanism that can effectively capture relationships among nodes.

D.1. Effect of Edge Pruning

Figure 7 presents an ablation study on *PuzzleCelebA*, where we vary the pruning rate during training. Increasing the pruning, i.e., reducing the graph size, has a minor effect on the final results.

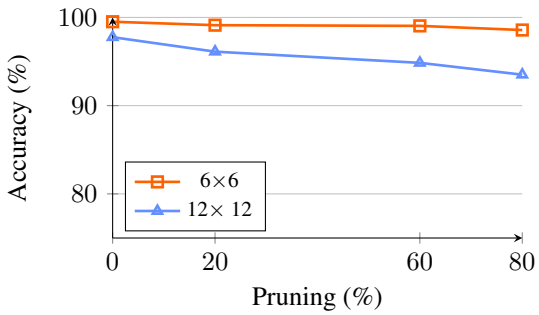

Figure 7. Ablation Sparse Attention Mechanism for Jigsaw puzzle solving on *PuzzleCelebA*.

E. Dataset Details

3D Reassembly Task. A 3D reassembly involves aligning fragments of a broken object into its original form, an essential task with applications in artifact preservation, digital heritage archiving, computer vision, robotics, and geometry processing. Despite its practical importance, the field has faced challenges due to the lack of suitable datasets for studying the natural fracture process. Existing datasets, such as PartNet [33], AutoMate [23], and JoinABLe [49], rely on semantic segmentation, failing to represent objects broken under natural, physically realistic conditions. Breaking Bad (BB) [37] fills this gap by simulating fractures using an algorithm that accounts for an object's most geometrically natural breaking patterns, thus creating a dataset that more realistically represents the challenges faced in fragments reassembly. **BB** contains approximately 10,000 meshes sourced from PartNet and Thingi10k. Each mesh includes 80 fractures, resulting in a total of 1,047,400 breakdown patterns. The dataset is divided into three subsets: *everyday*, *artifact*, and *other*. In this work, we focus on the *everyday* subset, as it is the commonly used dataset for evaluation in previous literature [50]. Qualitative examples can be found in the video attached to this Supplementary Material.

2D Reassembly Task. In this task, we evaluated DiffAssemble on two datasets: PuzzleCelebA and PuzzleWikiArts. Figure 8 shows some examples of inputs and reconstructions. More examples can be found in the video attached as Supplementary Material.

- *PuzzleCelebA* is based on CelebA-HQ [26] which contains 30K images of celebrities in High Definition (HD). Despite its superficial simplicity, this dataset poses significant challenges for puzzle-solving algorithms due to the inherent symmetry in human faces and often indistinct backgrounds. The dataset is divided in 80-20% train-test split, with 6,000 test puzzle permutations and randomly rotated patches.
- *PuzzleWikiArts* is based on WikiArts [46], and contains 63K images of paintings in HD. This dataset is particularly challenging due to very different content, artistic styles, and intricate patterns, which test the limits of puzzle-solving algorithms. The dataset is split into an 80-20% train-test ratio, resulting in 50k training images and 13k test puzzles across various grid sizes. It represents a more challenging dataset for puzzle solving as the paintings do not have a common pattern as in PuzzleCelebA (i.e. portraits).

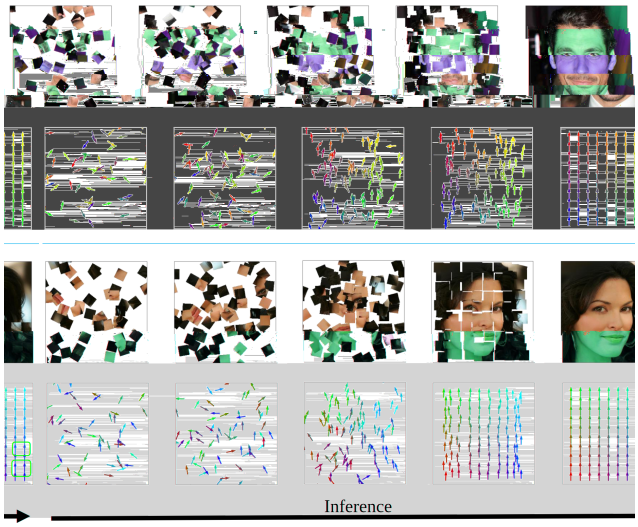

Figure 8. Qualitative results showing the diffusion process from random to solved puzzle. Each arrows correspond to one piece of the puzzle and its orientation indicate the orientation of the piece.

METHOD W/ % DEGREE	PuzzleCelebA							
	6x6	**8x8**	**10x10**	**12x12**	**14x14**	**16x16**	**18x18**	**20x20**
Degree 20%								
Classical dropout	91.60	57.08	82.32	50.18	74.43	25.40	61.45	28.35
Sparse Attention Mechanism	**92.37**	**59.45**	**87.67**	**54.07**	**83.11**	**31.07**	**73.88**	**32.97**
Degree 60%								
Classical dropout	**99.17**	72.93	**98.56**	94.07	98.53	46.48	**98.35**	92.51
Sparse Attention Mechanism	99.04	**73.91**	98.43	**94.35**	**98.70**	**48.26**	97.75	**93.29**
Degree 80%								
Classical dropout	99.15	76.45	98.75	95.87	98.58	51.51	98.14	94.93
Sparse Attention Mechanism	99.15	78.18	98.77	95.71	98.69	52.28	97.80	94.34

Table 7. Ablation Sparse Attention Mechanism for Jigsaw puzzle solving on *PuzzleCelebA*.