Exact Shape Correspondence via 2D Graph Convolution

Anonymous Author(s)
Affiliation
Address
email

Abstract

For exact 3D shape correspondence (matching or alignment), i.e., the task of matching each point on a shape to its exact corresponding point on the other shape (or to be more specific, matching at geodesic error 0), most existing methods do not perform well due to two main problems. First, on nearly-isometric shapes (i.e., low noise levels), most existing methods use the eigen-vectors (eigen-functions) of the Laplace Beltrami Operator (LBO) or other shape descriptors to update an initialized correspondence which is not exact, leading to an accumulation of update errors. Thus, though the final correspondence may generally be smooth, it is generally inexact. Second, on non-isometric shapes (noisy shapes), existing methods are generally not robust to noise as they usually assume near-isometry. In addition, existing methods that attempt to address the non-isometric shape problem (e.g., GRAMPA) are generally computationally expensive and do not generalise to nearly-isometric shapes. To address these two problems, we propose a 2D graph convolution-based framework called 2D-GEM. 2D-GEM is robust to noise on non-isometric shapes and with a few additional constraints, it also addresses the errors in the update on nearly-isometric shapes. We demonstrate the effectiveness of 2D-GEM by achieving a high accuracy of 90.5% at geodesic error 0 on the non-isometric benchmark SHREC16, i.e., TOPKIDS (while being much faster than GRAMPA), and on nearly-isometric benchmarks by achieving a high accuracy of 92.5% on TOSCA and 84.9% on SCAPE at geodesic error 0.

1 Introduction

3D shapes can undergo different types of deformations. Deformations can be rigid or non-rigid. Rigid deformations include rotations, translations, reflections, or any composition of these resulting in pose changes. They preserve geometric properties of the shape such as segment lengths, angles, volumes, sizes, geodesic distances and others. Non-rigid deformations do not preserve geometric properties of the shape, they include dilations, or a composition of dilations and rigid deformations (e.g., stretching, scaling). In this work we focus on non-rigid shapes that are either nearly-isometric (i.e., most geometric properties are preserved), or non-isometric (i.e., geometric properties are not preserved, and contain strong topological noise).

Non-rigid shape matching is a very important task in machine learning, shape analysis and computer vision. It has important applications such as shape registration, comparison, recognition, and retrieval. 3D shapes are usually represented as 2D manifolds embedded in $\mathbb{R}^3$. Given two three dimensional shapes $\mathcal{M}_1$ and $\mathcal{M}_2$ with n vertices each, the goal of non-rigid shape matching is to find a meaningful correspondence $\mathcal{T} : \mathcal{M}_1 \rightarrow \mathcal{M}_2$, where $\mathcal{T}$ is (a) bijective, (b) continuous in both directions in the sense that nearby points on $\mathcal{M}_1$ should be matched to nearby points on $\mathcal{M}_2$ (vice versa), and (c) similar points should be matched to each other [39]. In the discrete setting the correspondence (map or match) $\mathcal{T}$ can be represented as a permutation matrix $\mathbf{P} \in \{0, 1\}^{n \times n}$ [51, 4].

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

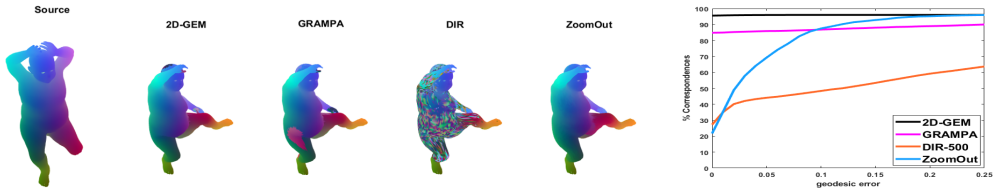
In more details, 3D shapes are usually represented using triangulations (e.g., the Delauney triangulation [18]) and xyz-coordinates representations in $\mathbb{R}^3$ [77, 39]. It is hard to utilize these xyz-coordinates and triangulations to directly perform 3D shape matching because of: (a) embedding ambiguities, i.e. shapes sharing the same metric (isometry) or very similar metrics (nearly-isometric) can have drastically different xyz-coordinates representations in $\mathbb{R}^3$ due to pose changes [77], and (b) non-isometric shapes with strong topological noise will have even more different xyz-coordinates and triangulations.

As such, these 3D-coordinates together with the triangulations are usually used to extract *point-wise descriptors* (which encode the similarity between points), and *pair-wise descriptors* (which encode global and/or local relations between pairs of points). Once these descriptors are built, the shape matching problem is reduced to an alignment problem over (a) the *point-wise descriptors*, or (b) the *pair-wise descriptors*, or (c) *both (joint alignment)*. The *joint alignment* problem can be written as:

$$\mathbf{P} = \arg\min_{\mathbf{P}\in\Pi} \mathbb{E}(\mathbf{P}) = \arg\min_{\mathbf{P}\in\Pi} \mathbb{F}(\mathbf{P}) + \alpha\mathbb{H}(\mathbf{P}), \tag{1}$$

where Π is the set of permutation matrices, $\mathbb{E}$ is the generalised energy, $\mathbb{F}$ is the energy over *point-wise descriptors*, $\mathbb{H}$ is the energy over *pair-wise descriptors*, and α is a trade-off parameter [39]. Several algorithms have been proposed to address point-wise alignment, [4, 67, 11, 52, 61], pair-wise alignment [18, 16], as well as joint alignment [76, 39, 51, 55]. Due to space constraints, see Appendix A for more details on the different alignments as well as related works.

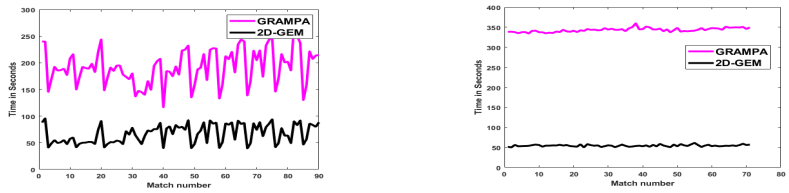
To measure the performance of an algorithm on the 3D shape matching task, the geodesic error is usually used as the error metric [18, 77]. Given that a point x on shape $\mathcal{M}_1$ is mapped by an algorithm to y as its correspondence on shape $\mathcal{M}_2$, the geodesic error measures how far the match y is from the true matching position for x on $\mathcal{M}_2$ i.e., y^* (given by the true map). When the map returned by algorithm is exact, it significantly improves performance and efficiency of other tasks such as registration, comparison, motion transfer, style transfer. However, most methods proposed in each of the aforementioned categories for 3D shape matching do not perform well at geodesic error 0 (i.e., the task of exact matching where y is the same point y^*). This is mainly due to two factors.



(a) Sample map on TOPKIDS (b) Plot of sample on TOPKIDS

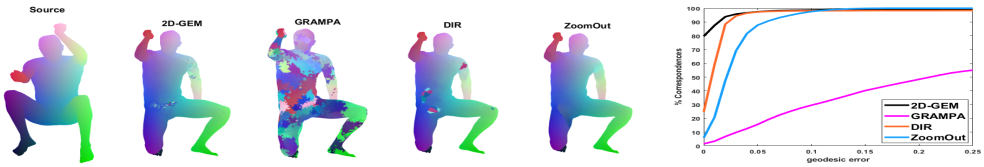
Figure 1: A non-isometric example. Illustrating the performance of 2D-GEM and other methods on a sample shape from the TOPKIDS dataset. For DIR we used DIR-500 with all LMD thresholds set to 5 since geodesic distances are not conserved on TOPKIDS (note that here, DIR performs much worse with lower thresholds). For ZoomOut we used 200 eigen-values and started from 20 with a step size of 1. For 2D-GEM and GRAMPA we followed the parameters in Section D.4

First, on non-isometric shapes, given that most existing methods assume near-isometry, they cannot handle noisy shapes (e.g., shapes with strong topological noise such as mesh "gluing" in areas of contact as in the TOPKIDS dataset [39]), since in this case geometric properties such as geodesic distances are not preserved and hence the shapes are not nearly isometric. An illustration can be seen in Figures 1(a) and 1(b), where the existing methods that assume near-isometry such as ZoomOut [51] and DIR [77] do not perform well at geodesic error 0 on the sample shape from TOPKIDS. They even produce a non-smooth map in the case of DIR. Moreover, those methods that try to address this noise are computationally very expensive and do not generalize well to nearly-isometric shapes, because for nearly-isometric shapes stronger constraints (such as preserving geodesic distances) are needed for the match. To illustrate this point, the spectral method GRAMPA is very expensive as it needs to compute all the eigen-values and eigen-vectors of the adjacency matrices obtained from the triangulations of each shape (see more details in Appendix B and time comparison with 2D-GEM in Figures 2(a) and 2(b)). Moreover, GRAMPA cannot generalize to cases of near-isometry as illustrated in Figure 3(a), which shows the matching of GRAMPA in an example from SCAPE, a nearly-isometric shape dataset).



(a) Times on TOPKIDS (b) Times on SCAPE

Figure 2: Time taken by 2D-GEM and GRAMPA for each of the 90 shapes on TOPKIDS and 71 on SCAPE.



(a) Sample map on SCAPE (b) Plot of sample on SCAPE

Figure 3: An isometric example. Illustrating the performance of 2D-GEM and other methods on a sample shape from the SCAPE dataset (using same settings as in figure 1).

Second, for nearly-isometric shapes, most existing methods attempt to first find an initial correspondence and then iteratively improve this initial correspondence using all points and the eigen-vectors corresponding to lower frequencies (eigen-values) of the LBO [4, 55, 51] or using other shape descriptors [61]. However, such an approach leads to an accumulation of errors in the correspondence, since the initial correspondence is mostly not accurate. This problem has been discussed in details by Xiang et al. [77], which attempted to address this via the Dual Iterative Refinement (DIR). DIR finds well matched points using the notion of Local Mapping Distortion (LMD) to identify points for which the geodesic distances are well preserved by the correspondence, and then use these points to update the correspondences of the other points via the notion of Functional maps [53]. Thus, DIR reduces the error accumulation. However, the DIR approach only works for nearly-isometric shapes and not for non-isometric shapes since it is based on the assumption of preserving geodesic distances which are not preserved in the case of non-isometries (see Figure 1(a), which shows that DIR does not perform well at geodesic 0 for non-isometric shapes). Finally, even on nearly-isometric shapes, DIR, though effective, requires a lot of eigen-vectors of the LBO in order to perform well at geodesic error 0. This can be seen in Figure 3(b), where using up to 500 eigen-vectors gives only 25% accuracy at geodesic error 0 on the sample shape from SCAPE, and even with up to 1000 eigen-vectors it only achieves 69.5% correspondence accuracy on SCAPE and a 59.8% on TOSCA at geodesic error 0 (as reported in the experiments in Section 5).

Given these problems, we propose to utilize 2D graph convolution for (a) non-isometric shape matching (since it is a de-noising operation), and (b) for isometric shape matching by further constraining our framework to preserve geodesic distances. To this end:

- we show how graph matching can be done via the 2D graph convolution, and re-define noise in the context of graph matching. We further show empirically that this approach can match the recent spectral method GRAMPA on Erdos Regny random graph matching, as well as outperform it in the presence of informative node features (see Appendix D),
- we propose a new 2D graph convolution-based approach to shape matching, called 2D-GEM, which is (a) robust to noise, and achieves high accuracy on exact matching of non-isometric shapes while being orders of magnitude faster than GRAMPA, and (b) with some additional constraints also achieves high accuracy on exact matching of nearly-isometric shapes.

2 2D graph Convolution for Joint Alignment

Here, we revisit graph convolution, show how the *joint alignment* problem can be done via 2D graph convolution, and finally we re-define noise in the context of graph matching.

2.1 Graph Convolution

From the Graph Signal Processing [64] perspective, graph convolution is regarded as a frequency filtering operator over node attributes. Given a graph, with node attribute matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, its normalized symmetric Laplacian (referred to as Laplacian in the rest of the paper) can be eigen-decomposed as $\mathbf{L} = \mathbf{U}\Lambda\mathbf{U}^T$, where $\Lambda = \text{diag}(\lambda_1, \cdots, \lambda_n) \in \mathbb{R}^{n \times n}$ are the eigen-values in increasing order, and $\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_n] \in \mathbb{R}^{n \times n}$ are the associated orthogonal eigen-vectors. The graph convolution operation applied to the graph signal $\mathbf{x}$ (a column of $\mathbf{X}$) consists of three steps [42, 22, 27]: (1) transform node attributes into the spectral domain (i.e., $\mathbf{z} = \mathbf{U}^T\mathbf{x}$); (2) remove attribute noise by scaling the spectrum with a function defined on eigen-values (i.e., $\bar{z}_i = g(\lambda_i)z_i$); (3) perform inverse transformation on the scaled spectrum (i.e., $\bar{\mathbf{x}} = \mathbf{U}\bar{\mathbf{z}}$), where $g(\lambda)$ is a scalar-valued function on the eigen-values λ. In short, graph convolution is able to remove the noisy signals by assigning smaller weight on higher eigen-value spectrum with a well defined function $g(\lambda)$. For convenience, we denote $g(\mathbf{L}) = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^T$, so that we have the convolution as $g * \mathbf{X} = g(\mathbf{L})\mathbf{X}$. We refer to $g(\mathbf{L})$ as the *convolution function* and $g(\mathbf{\Lambda})$ as the *filter function* or simply filter.

2.2 Joint Alignment via 2D graph Convolution

Following previous works [20, 68] and Appendix C, the graph convolution is invariant to permutation. As such, given two graphs $\mathcal{A}$ and $\mathcal{B}$ with graph convolution functions $\mathbb{M}_A = g(\mathbf{L}_A), \mathbb{M}_B = h(\mathbf{L}_B) \in \mathbb{R}^{n \times n}$, features $\mathbf{X}_A, \mathbf{X}_B \in \mathbb{R}^{n \times d}$, and having the functions g and h to be the same function, the *joint alignment* problem in Equation (1) can instead be performed as:

$$\min_{\mathbf{P}} ||\mathbf{P}\mathbb{M}_A\mathbf{X}_A - \mathbb{M}_B\mathbf{X}_B||_F^2, \tag{2}$$

see Appendix C for more details. We observe that (2) reduces to the Linear Assignment Problem (LAP):

$$\max_{\mathbf{P}} \mathbf{Tr}(\mathbf{P}\mathbb{M}_A\mathbf{X}_A\mathbf{X}_B^T\mathbb{M}_B), \tag{3}$$

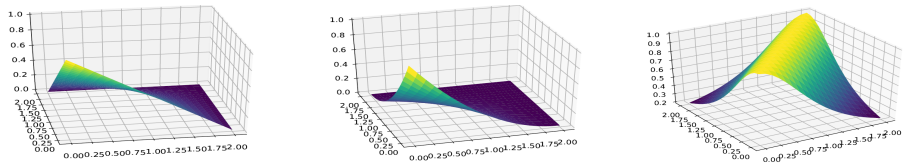
where $\mathbf{Tr}(\cdot)$ is the matrix trace operator, the convolution functions capture the pair-wise descriptors, and the features capture the point-wise descriptors. The first benefit of the convolution is to help do the joint alignment as a Linear Assignment Problem (LAP), which is much cheaper than the traditional Quadratic Assignment Problem (QAP) [21] used for graph matching. Note that, $\mathbb{M}_A = \mathbf{U}g(\Lambda)\mathbf{U}^T = \sum_i g(\lambda_i)\mathbf{u}_i\mathbf{u}_i^T$ and $\mathbb{M}_B = \mathbf{V}h(\mathbf{M})\mathbf{V}^T = \sum_j h(\mu_j)\mathbf{v}_j\mathbf{v}_j^T$, where $\mathbf{U}, \mathbf{V}$ are the orthogonal eigen-vectors of $\mathbb{M}_A$ and $\mathbb{M}_B$, and $\Lambda, \mathbf{M}$ are the diagonal matrices of their respective eigen-values. Given these, one can rewrite our cost matrix $\mathbf{C} = \mathbb{M}_A\mathbf{X}_A\mathbf{X}_B^T\mathbb{M}_B$ in (3) as:

$$\mathbf{C} = \sum_{i,j} g(\lambda_i)h(\mu_j)\mathbf{u}_i\mathbf{u}_i^T\mathbf{X}_A\mathbf{X}_B^T\mathbf{v}_j\mathbf{v}_j^T, \tag{4}$$

When node features are not available, one can initialize all nodes with the same features in (3), i.e., $\mathbf{X}_A = \mathbf{J}_A \in \mathbb{R}^{n_1 \times d}$, and $\mathbf{X}_B = \mathbf{J}_B \in \mathbb{R}^{n_2 \times d}$, where n_1 may or may not be equal to n_2 for different size graphs, and $\mathbf{J}_A, \mathbf{J}_B$ are all one rectangular matrices. This is equivalent to assuming that nodes have the same probability of being matched to each other. Moreover, to save cost in the case when there is no feature, one could simply initialize the similarity matrix $\mathbf{X}_A\mathbf{X}_B^T = \mathbf{J}_{A,B} \in \mathbb{R}^{n_1 \times n_2}$ (given that $\mathbf{J}_A\mathbf{J}_B^T = \mathbf{J}_{A,B} \in \mathbb{R}^{n_1 \times n_2}$, where $\mathbf{J}_{A,B}$ is a scalar matrix with constant $[\mathbf{J}_{A,B}]_{ij} = \sum_{i=1}^d 1 = d$) for all entries. In the case where there is no feature, (4) becomes:

$$\mathbf{C} = \sum_{i,j} g(\lambda_i)h(\mu_j)\mathbf{u}_i\mathbf{u}_i^T\mathbf{J}_{A,B}\mathbf{v}_j\mathbf{v}_j^T. \tag{5}$$

Our cost matrices in equations (4) and (5) can be seen as 2D graph convolution on the 2D graph signals $\mathbf{X}_A\mathbf{X}_B^T$ (or $\mathbf{J}_{A,B}$ in the case of no features) [43], where we: (1) transform the 2D graph signals into the spectral domain to obtain the 2D graph spectrum (i.e., $\mathbf{Z} = \mathbf{U}^T\mathbf{X}_A\mathbf{X}_B^T\mathbf{V}$ or $\mathbf{Z} = \mathbf{U}^T\mathbf{J}_{A,B}^T\mathbf{V}$) [43]; (2) remove noise (de-noise) from the 2D graph signals by scaling the 2D spectrum with a function defined on each pair of eigen-values (i.e., $\bar{z}_{ij} = w(\lambda_i, \mu_j)z_{ij}$); and (3) perform inverse transformation on the scaled 2D spectrum ($\mathbf{C} = \mathbf{U}\bar{\mathbf{Z}}\mathbf{V}^T$). Thus, our cost matrices are 2D filtering operation on the 2D graph signals, where $w(\lambda_i, \mu_j) = g(\lambda_i)h(\mu_j)$ is the 2D filter (see Figures 4(a), 4(b) and 4(c) for different 2D filters).



(a) 1-hop low-pass (b) 2-hop low-pass (c) redefining noise

Figure 4: 2D graph filters. Figure 4(a) uses the convolution functions $g(\mathbf{L}_A) = g(\mathbf{L}_B) = (\mathbf{I} - \frac{1}{2}\mathbf{L})$, Figure 4(b) uses the convolution functions $g(\mathbf{L}_A) = g(\mathbf{L}_B) = (\mathbf{I} - \frac{1}{2}\mathbf{L})^2$, and Figure 4(c) uses the noise robust filter defined in Section 3 with $e = 1$.

3 Redefining Noise in the Context of Graph Matching

In this section, we redefine noise in the context of graph matching and propose a suitable filter for graph convolution with regards to the graph matching task. It can be shown that graph convolution is a solution to the graph signal de-noising problem [22]. Formally, assuming that (1) the ground-truth node features $\hat{\mathbf{X}}$ are smooth w.r.t. a graph $\mathcal{G}$ with adjacency $\mathbf{A}$ and Laplacian $\mathbf{L}$, and that (2) the node feature noise has small magnitude, then the total variation of $\hat{\mathbf{X}}$ should be small by assumption (1), and $||\hat{\mathbf{X}} - \mathbf{X}||$ can be upper-bounded by assumption (2). Thus, we can model the problem of recovering the ground-truth signal as:

$$\hat{\mathbf{X}}^* = \arg\min_{\hat{\mathbf{X}}} \mathbf{Tr}\left(\hat{\mathbf{X}}^\top \mathbf{L} \hat{\mathbf{X}}\right) \qquad \text{s.t.} ||\hat{\mathbf{X}} - \mathbf{X}||_F^2 \leq \epsilon_1, \tag{6}$$

It can clearly be observed that $\hat{\mathbf{X}}^* = (\mathbf{I} + \alpha\mathbf{L})^{-1}\mathbf{X}$, where α is a hyperparameter. This is a graph convolution with graph convolution function given by $(\mathbf{I} + \alpha\mathbf{L})^{-1}$. This function can be decomposed as $\mathbf{U}(\mathbf{I} + \alpha\Lambda)^{-1}\mathbf{U}^T$. As well studied in [42, 75, 33], one can see that the filter $(\mathbf{I} + \alpha\Lambda)^{-1}$ is decaying, i.e., $g(\lambda_i)$ is lower for higher λ_i, which will mostly reduce the amplitude z_i of graph signals $\mathbf{u}_i$ belonging to larger eigen-values λ_i. Thus, one can conclude that noise corresponds to higher order frequencies (eigen-values). This kind of filter is commonly known as a low-pass filter [42]. It can be shown that for the 2D graph convolution, filters of the form $w(\lambda_i, \mu_j) = g(\lambda_i)h(\mu_i)$, where $g(\lambda_i) = (1 - \alpha\lambda_i)^k$ and $h(\mu_j) = (1 - \alpha\mu_j)^k$ will result in a 2D low-pass filter as well (see Figures 4(a) and 4(b)). These filters are referred to as k-hop low-pass filters (aggregating information from the k-hop neighborhood of a node).

However, in the context of graph matching, graphs are isomorphic when they have the same eigen-value, and in this case we want to align the eigen-vectors since two graphs $\mathcal{A}$ and $\mathcal{B}$ are isomorphic iff their eigen-vectors are permutations of each other up to a sign [69, 34].

This leads us to redefine noise in the graph matching context. Given two graphs $\mathcal{A}$ and $\mathcal{B}$, with convolution functions $\mathbb{M}_A$, $\mathbb{M}_B$ whose eigen-values are λ_i and μ_j respectively, noise in graph matching corresponds to 2D graph signals $z_{i,j} = \mathbf{u}_i^T \mathbf{X}_A \mathbf{X}_B^T \mathbf{v}_j$ or $z_{i,j} = \mathbf{u}_i^T \mathbf{J}_{A,B} \mathbf{v}_j$ such that $\lambda_i \neq \mu_j$, i.e., eigen-values that differ. Thus, we should use a filter that kills off these signals corresponding to eigen-values that do not match. We choose the filter $w(\lambda_i, \mu_j, e) = \frac{1}{(\lambda_i^e - \mu_j^e)^2 + 1}$ instead of $g(\lambda_i)h(\mu_j)$. This filter exactly kills off these signals corresponding to eigen-values that differ (see Figure 4(c) to see an illustration of its action on signals). A note here is that the amplitude and the width of the filter response in Figure 4(c) depends on the spectra of the graphs to be matched. For simplicity, we denote this filter as $w(\lambda_i, \mu_j)$ in the rest of the paper. Thus, a noise robust 2D graph convolution based cost matrix for graph matching is given as:

$$\mathbf{C} = \sum_{i,j} w(\lambda_i, \mu_j) \mathbf{u}_i \mathbf{u}_i^T \mathbf{X}_A \mathbf{X}_B^T \mathbf{v}_j \mathbf{v}_j^T. \tag{7}$$

Moreover, to reduce the time complexity, one may use the first k eigen-values $\Lambda_k = diag(\lambda_i, \cdots, \lambda_k)$, $\mathbf{M}_k = diag(\mu_i, \cdots, \mu_k)$ and eigen-vectors $\mathbf{U}_k = [\mathbf{u}_i, \cdots, \mathbf{u}_k]$, $\mathbf{V}_k = [\mathbf{v}_i, \cdots, \mathbf{v}_k]$, instead of the full spectrum. Doing so, the noise robust 2D graph convolution based cost matrix for graph matching becomes:

$$\mathbf{C} = \sum_{i,j}^{k} w(\lambda_i, \mu_j) \mathbf{u}_i \mathbf{u}_i^T \mathbf{X}_A \mathbf{X}_B^T \mathbf{v}_j \mathbf{v}_j^T. \tag{8}$$

Algorithm 1 : 2D-GEM

1. **input** An initial correspondence $\mathbf{P}^0$, $\mathbf{U}_k$, $\mathbf{V}_k, \mathbf{\Lambda}_k, \mathbf{M}_k, e$, maximum iteration $iter$, and the LMD error $\{\epsilon_i\}_{i=1}^{iter}$ thresholds.

while $0 \leq i \leq iter$ **do**

2. use one round of the 2-Hop algorithm. i.e., set $t = 1$ in Algorithm 2
3. use the LMD to locate well anchored pairs (landmarks) l
4. use these well anchored pairs to update the 2D graph spectrum as in (9)
5. use the updated spectrum for updating the correspondence of poorly matched points p: $\max_{\mathbf{P}_p} \mathbf{Tr}(\mathbf{P}_p \mathbf{C}_p)$, with $\mathbf{C}_p$ as in (10)

end while

Algorithm 2 : 2-Hop

1. **input** An initial correspondence $\mathbf{P}$, 2-hop adjacency matrices $\mathbf{A}_2$, $\mathbf{B}_2$, maximum iteration t.

while $0 \leq$ i $\leq$ t **do**

2. $\mathbf{P}^{i+1} = GMWM(\mathbf{W})$, where $\mathbf{W} = \mathbf{A}_2 \mathbf{P}^i \mathbf{B}_2$

end while

3. **return** $\mathbf{P}^t$

Due to space constraints, we refer the reader to Appendix D for the experimental validation of the 2D graph convolution approach for graph matching.

4 Our Algorithm

Having formulated the 2D graph convolution-based approach for graph matching as well as redefined noise in the context of graph matching, we now formulate 2D-GEM (2D Graph convolution-based framework for attempting Exact shape Matching) for non-isometric and nearly-isometric 3D shape matching.

4.1 2D-GEM

For our framework, given the xyz coordinates of two shapes $\mathcal{M}_1$ and $\mathcal{M}_2$, we build suitable point-wise descriptors $\mathbf{X}_A$ and $\mathbf{X}_B$, as well as graph laplacians $\mathbf{L}_A$ and $\mathbf{L}_B$ using their triangulations. We then initialize the correspondence using the point-wise descriptors. Then at each iteration, we first follow GRAMPA and update the correspondence using the 2-hop neighborhood graph matching algorithm [81], which makes use of the Greedy Maximum Weight Matching (GMWM) with computational complexity $O(n^2 \log n)$ (see Algorithm 2 and Appendix B). The 2-hop adjacency matrices $\mathbf{A}_2$ and $\mathbf{B}_2$ of the shapes $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively, are such that $[\mathbf{A}_2]_{i,j} = 1$ and $[\mathbf{B}_2]_{i,j} = 1$ if i and j are 2-hop neighbors, and 0 otherwise. Second, we use the notion of conservation of geodesic distances [61] to further constrain 2D-GEM, we capture this via LMD[77]. Finally, we adopt the 2D graph convolution in Equation (8) to filter the correspondence $\mathbf{P}$ rather than the feature similarity matrix $\mathbf{X}_A \mathbf{X}_B^T$ in Equation (8), i.e., we use $\mathbf{P}$ rather than $\mathbf{X}_A \mathbf{X}_B^T$ as our 2D graph signals. This can be seen as de-noising the correspondence at each iteration following the state-of-the-art Kernel-Matching method [39].

In details (see Algorithm 1), we initialize the correspondence via SHOT[67], and then at each iteration ($iter$):

- we update the correspondence using the 2-Hop Algorithm (2) with $t = 1$,
- we find l anchor pairs and p non-anchor pairs via LMD (see Section 4.2) and use them to update the 2D graph spectrum as

$$\mathbf{Z}_l = \mathbf{U}_k^T[l] \mathbf{P}_l \mathbf{V}_k[l], \tag{9}$$

where $\mathbf{U}_k[l], \mathbf{V}_k[l] \in \mathbb{R}^{l \times k}$ and $\mathbf{P}_l \in \mathbb{R}^{l \times l}$ are the eigen-vectors and correspondence of the l-anchor pairs (landmarks) found, and $\mathbf{Z}_l$ is the updated $k \times k$ 2D graph spectrum.

- we use the updated spectrum for updating the correspondence of poorly matched points as $\max_{\mathbf{P}_p} \mathbf{Tr}(\mathbf{P}_p \mathbf{C}_p)$, where

$$\mathbf{C}_p = \sum_{i,j}^{k} w(\lambda_i, \mu_j) \mathbf{u}_i[p][\mathbf{Z}_l]_{i,j} \mathbf{v}_j^T[p], \tag{10}$$

  and $\mathbf{u}_i[p], \mathbf{v}_j[p] \in \mathbb{R}^p$ are the eigen-vectors of the non-landmark points p.

Computation complexity. The complexity for computing k leading eigen-vectors of a $n \times n$ sparse matrix is $O(kn \log n)$. The complexity for checking LMD is $O(n)$ following DIR [77]. The GMWM has a computational complexity of $O(n^2 \log n)$. Thus, our method is of complexity $O((iter)n^2 \log n) = O(n^2 \log n)$, since $iter$ is usually small, e.g., 10 or 60 in our experiments.

4.2 Local Mapping Distortion (LMD)

Here, we describe the LMD employed by 2D-GEM to get landmarks. The LMD was introduced by [77] and [76] for nearly-isometric shapes. Let $T : \mathcal{M}_1 \to \mathcal{M}_2$ be a map function between two shapes. The LMD of the map T at the point x_i is given as follows:

$$\mathcal{D}_\gamma(T)(x_i) = \frac{\sum_{x_j \in \mathcal{B}_\gamma(x_i)} \mathcal{A}_1(j) DE_T(x_i, x_j)}{\sum_{x_j \in \mathcal{B}_\gamma(x_i)} \mathcal{A}_1(j)}, \tag{11}$$

where $\mathcal{B}_\gamma(x) = \{y \in \mathcal{M}_1 \mid d_{\mathcal{M}_1}(x, y) \leq \gamma\}$ is the γ-geodesic ball of x, $\mathcal{A}_1$ is the area element of the mesh of shape $\mathcal{M}_1$, and $DE_T(x, y) = |d_{\mathcal{M}_1}(x, y) - d_{\mathcal{M}_2}(T(x), T(y))|/\gamma$ represents a pair-wise distance distortion of mapping nearby points x_i and x_j to $T(x_i)$ and $T(x_j)$. A smaller value of $\mathcal{D}_\gamma(T)(x_i)$ means a better map continuity of T at the point x_i, in other words, the local distance at the point x_i is well preserved. Based on the above definition of LMD, one can check that if T is an isometric map, then $\mathcal{D}_\gamma(T)(x) = 0, \forall x \in \mathcal{M}_1, \gamma > 0$. Conversely, if $\mathcal{D}_\gamma(T)(x) = 0, \forall x \in \mathcal{M}_1$ for some $\gamma > 0$, then T is isometric. We use the LMD to find well matched pairs, i.e., pairs for which $\{(x_i, T(x_i)) | \mathcal{D}_\gamma(T)(x_i) \leq \epsilon_i\}$, where ϵ_i is a threshold. We call them anchor points (landmarks l), and others non-anchor points (poor-points p). We use the landmarks to update the 2D graph spectrum $\mathbf{S}_{kl}$, which we use to update the correspondence of non-landmark points (see Section 4.1). It is important to note that at every iteration we gradually decrease the threshold following Xiang et al. [77], and hence not only is the correspondence T updated, but also the landmarks l. For non-isometric shapes we relax this constraint by setting the thresholds ϵ_i arbitrarily large, such as 100 for 2D-GEM in the experiments (see Section 5.5).

5 Experiments

We report experimental results that validate the effectiveness and efficiency of 2D-GEM on exact matching of non-isometric and nearly-isometric 3D shapes.

5.1 Experimental Set-up

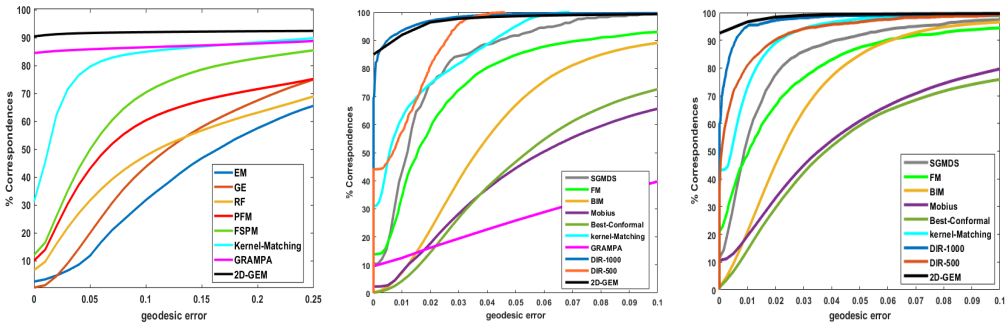
Time comparison between 2D-GEM and GRAMPA is conducted in Matlab on a Windows 10 system with 16GB RAM and Intel(R) i5 11400 CPU @ 2.60-4.4GHz.

5.2 Datasets

We conduct experiments to evaluate the performance of our method on two nearly isometric benchmark datasets TOSCA [10], and SCAPE [5], as well as a benchmark non-isometric dataset SHREC'16 (TOPKIDS) [39]. TOSCA consists of 80 shapes in 8 different categories (human and animal shapes) with vertex numbers ranging from 4k to 50k. SCAPE has 71 shapes (12,500 vertices for each) of the same person with different poses. TOPKIDS contains 25 shapes of the same class with up to 12K vertices, undergoing near-isometric deformations in addition to large topological shortcuts (noise).

5.3 Evaluation Metrics

We use the geodesic error as our error metric [18]. Given that the correspondence of an algorithm maps $x \in \mathcal{M}_1$ to $y \in \mathcal{M}_2$, and the true correspondence maps x to y^*, the geodesic error is defined as



(a) TOPKIDS (b) SCAPE (c) TOSCA

Figure 5: Performance comparison on TOPKIDS 5(a), SCAPE 5(b), and TOSCA 5(c).

$e(x) = \frac{d_{\mathcal{M}_2}(y,y^*)}{diam(\mathcal{M}_2)}$, where $d_{\mathcal{M}_2}$ denotes the geodesic distance on $\mathcal{M}_2$, and $diam(\mathcal{M}_2)$ is the geodesic diameter of $\mathcal{M}_2$.

5.4 Baselines

For non-isometric shape matching, i.e., for matching on the TOPKIDS dataset, we compare 2D-GEM with the following methods: EM [59], GE [38], RF [58], PFM [57], FSPM [46], Kernel-Matching [39], and GRAMPA [18].

For near-isometric shape matching, i.e., for matching on the SCAPE and TOSCA datasets, we compare 2D-GEM to the following methods: SGMDS[2], FM[53], BIM[31], Mobius[45], Best-Conformal [31], Kernel-Matching [39], DIR-500 [77] which uses 500 eigen-vectors, DIR-1000 [77] which uses 1000 eigen-vectors, and GRAMPA (on SCAPE, not on TOSCA due to its time complexity and the fact that 2D-GEM significantly outperforms it as shown on TOPKIDS and SCAPE).

5.5 Parameter Settings

For 2D-GEM, we initialize the correspondence by the GMWM on SHOT [67], and use $e = 10$ (see Appendix D for more analysis on e). For nearly-isometric shapes (TOSCA and SCAPE), we use 500 eigen-vectors of the LBO. Following Xiang et al. [77], we use the second ring neighborhood size for LMD criterion and set the maximum iteration number to $iter = 10$. We also set the LMD thresholds, we use $[0.5280, 0.4560, 0.3840, 0.3120, 0.2400, 0.1680, 0.1680, 0.1680, 0.1680, 0.1680]$. For non-isometric shapes (TOPKIDS), we set the thresholds to 100 for all iterations, use $iter = 60$, and 10 LBO eigen-vectors.

For other baselines, we report their results as reported in their papers i.e., we use their curves. Except for GRAMPA where we re-run using the 2-hop improvement algorithm instead which performs much better than the 1-hop, i.e., the adjacency. For fair comparison with 2D-GEM , we set $t = 60$ for this improvement on GRAMPA.

5.6 Performance Analysis

Comparison on non-isometric shapes. [1] GRAMPA is the state-of-the-art non-isometric shape matching method that achieves an excellent performance on the exact matching task on the noisy dataset TOPKIDS (Figure 5(a)). It achieves 84.6% on this dataset at geodesic error 0. This shows that GRAMPA is very robust to noise as expected (note that this performance is much higher than the 60% reported in their original paper because we used the 2-hop adjacency instead of the 1-hop for refinement as explained in Appendix B.2). However, our method 2D-GEM further outperforms GRAMPA and all other methods, including GRAMPA at geodesic error 0. As reported in Figure 5(a), where 2D-GEM achieves a 90.5% accuracy. Moreover, Figures 2(a) and 2(b) show that 2D-GEM is significantly faster than GRAMPA since 2D-GEM only needs to compute a subset of the eigen-vectors, unlike GRAMPA which needs the whole spectrum and thus has a complexity of $O(n^3)$.

[1] Ablation studies on 2D-GEM in Appendix E.

Comparison on nearly-isometric shapes. DIR is the state-of-the-art model for nearly-isometric shape matching. DIR outperforms all the other baselines except 2D-GEM. The better DIR-1000 using 1000 eigen-vectors achieves 69.5% accuracy at geodesic error 0 on SCAPE, and 59.8% accuracy at geodesic error 0 on TOSCA. Using the LMD proposed for the DIR model by Xiang et al. [77] for extra constraints on 2D-GEM, 2D-GEM not only generalizes to nearly-isometric shapes, but it achieves the state-of-the-art results using only 500 eigen-vectors, outperforming even DIR-1000. 2D-GEM achieves 92.5% accuracy (outperforming DIR-1000 by more than 40%) at geodesic error 0 on the TOSCA dataset, as well as 84.9% accuracy on the SCAPE dataset (outperforming DIR-1000 by almost 20%) . Note that these 2D-GEM's accuracy improves using more eigen-vectors or increasing $iter$, but this also increases the running time. Moreover, 2D-GEM's gain over GRAMPA (the state-of-the-art non-isometric shape matching method) can also be seen to extend to nearly-isometric shapes. As noted earlier in Section 1, GRAMPA does not generalize to nearly-isometric shapes since the assumptions for nearly-isometric shape matching are different from those of non-sometric shape matching given that isometries preserve geometric properties which are lost by non-isometries. This is seen by its poor performance on the SCAPE dataset (Figure 5(b), where even for exact matching, i.e., at geodesic error 0, it only achieves around 10%). In contrast, 2D-GEM (using the LMD for extra constraints) generalizes to nearly-isometric shapes and achieves the state-of-the-art performance. We do not attempt to use the LMD on GRAMPA since (1) it is not clear how to use GRAMPA in an iterative way as it is usually only used as an initialization for other algorithms (see Appendix B), (2) 2D-GEM outperforms GRAMPA even without the LMD as shown on the noisy TOPKIDS dataset (Figure 5(a)), and (3) 2D-GEM with or without the LMD is still faster than GRAMPA without the LMD (see Figures 2(a) and 2(b)).

5.7 Limitations of 2D-GEM

There are three main limitations to this work. First, for nearly-isometric shapes (shown in Figure 3(a)), 2D-GEM sometimes produces maps that are not continuous (smooth), though much more accurate than the other baselines. Second, although the time complexity $O(n^2 \log n)$ of 2D-GEM is much lower than the $O(n^3)$ of GRAMPA, it is much worse than the $O(n \log n)$ of other methods such as ZoomOut and DIR. Lastly, unlike GRAMPA, we do not provide rigorous theoretical exact recovery guarantees for 2D-GEM on random graph matching.

6 Conclusions

We presented how the 2D graph convolution could be used for graph matching. Using this 2D graph convolution, we re-defined noise in the context of graph matching and proposed a model (2D-GEM) that addresses the task of exact 3D shape matching on non-isometric (noisy) datasets as well as on nearly-isometric datasets. We then showed that 2D-GEM significantly outperforms existing methods on both non-isometric shapes and nearly-isometric shapes. For future work, we propose to address two key limitations of 2D-GEM, i.e., further reducing the time complexity of $O(n^2 \log n)$ and providing theoretical guarantees on its exact recovery performance. Moreover, we also propose to explore the generalization of 2D-GEM for correspondence tasks that allow using graph representation such as partial shape matching, image matching, and network de-anonymization.

7 Broader Impact

Given that our method needs no training, is unsupervised and based on intrinsic shape properties, it can positively benefit the industry and society as a whole in that; (a) it may be easily generalized, and most importantly (b) it needs no human labels nor training, which can help reduce the cost and time taken for the shape correspondence problems in different crucial fields such as medicine, and computer graphics.

References

[1] Yonathan Aflalo, Alexander Bronstein, and Ron Kimmel. On convex relaxation of graph isomorphism. *Proceedings of the National Academy of Sciences*, 112(10):2942–2947, 2015. ISSN 0027-8424. doi: 10.1073/pnas.1401651112.

[2] Yonathan Aflalo, Anastasia Dubrovina, and Ron Kimmel. Spectral generalized multi-dimensional scaling. *International Journal of Computer Vision*, 118:380–392, 2016.

[3] H.A. Almohamad and S.O. Duffuaa. A linear programming approach for the weighted graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5): 522–525, 1993. doi: 10.1109/34.211474.

[4] B. Amberg, S. Romdhani, and T. Vetter. Optimal step nonrigid icp algorithms for surface registration. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[5] Dragomir Anguelov, Praveen Srinivasan, Hoi-Cheung Pang, Daphne Koller, Sebastian Thrun, and James Davis. The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In *Advances in neural information processing systems*, pages 33–40, 2005.

[6] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1626–1633, 2011. doi: 10.1109/ICCVW.2011.6130444.

[7] László Babai, D. Yu. Grigoryev, and David M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82, page 310–324, New York, NY, USA, 1982. Association for Computing Machinery. ISBN 0897910702. doi: 10.1145/800070.802206.

[8] Yunsheng Bai, Hao Ding, Yizhou Sun, and Wei Wang. Convolutional set matching for graph similarity. In *NeurIPS Workshop on Relational Representation Learning*, 2018.

[9] Yunsheng Bai, Haoyang Ding, Song Bian, Ting Chen, Yizhou Sun, and Wei Wang. Simgnn: A neural network approach to fast graph similarity computation. *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019.

[10] Alexander M Bronstein, Michael M Bronstein, and Ron Kimmel. *Numerical geometry of non-rigid shapes*. Springer Science & Business Media, 2008.

[11] M. M. Bronstein and I. Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1704–1711, 2010.

[12] Donatello Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. Thirty years of graph matching in pattern recognition. *IJPRAI*, 18:265–298, 05 2004. doi: 10.1142/S0218001404003228.

[13] Chris Ding, Tao Li, and Michael I. Jordan. Nonnegative matrix factorization for combinatorial optimization: Spectral clustering, graph matching, and clique finding. In *2008 Eighth IEEE International Conference on Data Mining*, pages 183–192, 2008. doi: 10.1109/ICDM.2008.130.

[14] Jian Ding, Zongming Ma, Yihong Wu, and Jiaming Xu. Efficient random graph matching via degree profiles. *Probability Theory and Related Fields*, 179:29–115, 2020.

[15] B. L. Douglas. The weisfeiler-lehman method and graph isomorphism testing. *arXiv: Combinatorics*, 2011.

[16] Nadav Dym, Haggai Maron, and Yaron Lipman. Ds++: A flexible, scalable and provably tight relaxation for matching problems. *ACM Trans. Graph.*, 36(6), November 2017. ISSN 0730-0301. doi: 10.1145/3130800.3130826.

[17] Frank Emmert-Streib, Matthias Dehmer, and Yongtang Shi. Fifty years of graph matching, network alignment and network comparison. *Information Sciences*, 346-347:180–197, 02 2016. doi: 10.1016/j.ins.2016.01.074.

[18] Zhou Fan, Cheng Mao, Yihong Wu, and Jiaming Xu. Spectral graph matching and regularized quadratic relaxations: Algorithm and theory. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 2985–2995. PMLR, 13–18 Jul 2020.

[19] Soheil Feizi, Gerald Quon, Mariana Recamonde-Mendoza, Muriel Medard, Manolis Kellis, and Ali Jadbabaie. Spectral alignment of graphs. *IEEE Transactions on Network Science and Engineering*, 7(3):1182–1197, jul 2020. ISSN 2327-4697. doi: 10.1109/TNSE.2019.2913233.

[20] Matthias Fey, Jan E. Lenssen, Christopher Morris, Jonathan Masci, and Nils M. Kriege. Deep graph matching consensus. In *International Conference on Learning Representations*, 2020.

[21] Gerd Finke, Rainer E. Burkard, and Franz Rendl. Quadratic assignment problems. In Silvano Martello, Gilbert Laporte, Michel Minoux, and Celso Ribeiro, editors, *Surveys in Combinatorial Optimization*, volume 132 of *North-Holland Mathematics Studies*, pages 61–82. North-Holland, 1987. doi: https://doi.org/10.1016/S0304-0208(08)73232-8.

[22] Guoji Fu, Yifan Hou, Jian Zhang, Kaili Ma, Barakeel Fanseu Kamhoua, and James Cheng. Understanding graph neural networks from graph signal denoising perspectives. *ArXiv*, abs/2006.04386, 2020.

[23] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996. doi: 10.1109/34.491619.

[24] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

[25] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. Regal: Representation learning-based graph alignment. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 117–126, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450360142. doi: 10.1145/3269206.3271788.

[26] Ruqi Huang, Jing Ren, Peter Wonka, and Maks Ovsjanikov. Consistent zoomout: Efficient spectral map synchronization. In *Proc. SGP*, volume 39, 2020.

[27] Barakeel Fanseu Kamhoua, Lin Zhang, Kaili Ma, James Cheng, Bo Li, and Bo Han. Hypergraph convolution based attributed hypergraph clustering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021.

[28] Ehsan Kazemi and Matthias Grossglauser. On the structure and efficient computation of isorank node similarities. *ArXiv*, abs/1602.00668, 2016.

[29] Ehsan Kazemi, Seyed Hamed Hassani, M. Grossglauser, and H. Modarres. Proper: global protein interaction network alignment through percolation matching. *BMC Bioinformatics*, 17, 2016.

[30] Itay Kezurer, Shahar Z. Kovalsky, Ronen Basri, and Yaron Lipman. Tight relaxation of quadratic matching. *Computer Graphics Forum*, 34(5):115–128, 2015. doi: https://doi.org/10.1111/cgf.12701.

[31] Vladimir G. Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. *ACM Trans. Graph.*, 30(4), July 2011. ISSN 0730-0301. doi: 10.1145/2010324.1964974.

[32] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*, 2016.

[33] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[34] Stefan Klus and Tuhin Sahai. A spectral assignment approach for the graph isomorphism problem. *ArXiv*, abs/1411.0969, 2014.

[35] Oliver Koch, Nils M. Kriege, and Lina Humbeck. Chemical similarity and substructure searches. In *Encyclopedia of Bioinformatics and Computational Biology*, 2019.

[36] Giorgos Kollias, Shahin Mohammadi, and Ananth Grama. Network similarity decomposition (nsd): A fast and scalable approach to network alignment. *IEEE Transactions on Knowledge and Data Engineering*, 24(12):2232–2243, 2012. doi: 10.1109/TKDE.2011.174.

[37] Danai Koutra, Hanghang Tong, and David Lubensky. Big-align: Fast bipartite graph alignment. In *2013 IEEE 13th International Conference on Data Mining*, pages 389–398, 2013. doi: 10.1109/ICDM.2013.152.

[38] Z. Lähner, E. Rodolà, M. M. Bronstein, D. Cremers, O. Burghard, L. Cosmo, A. Dieckmann, R. Klein, and Y. Sahillioğlu. Matching of deformable shapes with topological noise. In *Proceedings of the Eurographics 2016 Workshop on 3D Object Retrieval*, 3DOR '16, page 55–60, Goslar, DEU, 2016. Eurographics Association. ISBN 9783038680048.

[39] Zorah Lähner, Matthias Vestner, Amit Boyarski, Or Litany, Ron Slossberg, Tal Remez, Emanuele Rodolà, Alexander M. Bronstein, Michael M. Bronstein, Ron Kimmel, and Daniel Cremers. Efficient deformable shape correspondence via kernel matching. *2017 International Conference on 3D Vision (3DV)*, pages 517–526, 2017.

[40] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1482–1489 Vol. 2, 2005. doi: 10.1109/ICCV.2005.20.

[41] Marius Leordeanu. *Spectral Graph Matching, Learning, and Inference for Computer Vision*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, July 2009.

[42] Qimai Li, Xiao-Ming Wu, Han Liu, Xiaotong Zhang, and Zhichao Guan. Label efficient semi-supervised learning via graph filtering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9582–9591, 2019.

[43] Qimai Li, Xiaotong Zhang, Han Liu, Quanyu Dai, and Xiao-Ming Wu. Dimensionwise separable 2-d graph convolution for unsupervised and semi-supervised learning on graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '21, page 953–963, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383325. doi: 10.1145/3447548.3467413.

[44] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects. In *ICML*, 2019.

[45] Yaron Lipman and Thomas Funkhouser. Möbius voting for surface correspondence. *ACM Trans. Graph.*, 28(3), July 2009. ISSN 0730-0301. doi: 10.1145/1531326.1531378.

[46] O. Litany, E. Rodolà, A. M. Bronstein, and M. M. Bronstein. Fully spectral partial shape matching. *Computer Graphics Forum*, 36(2):247–258, 2017. doi: https://doi.org/10.1111/cgf.13123.

[47] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1996–2003, 2009. doi: 10.1109/CVPR.2009.5206744.

[48] Vince Lyzinski, Donniell E. Fishkind, and Carey E. Priebe. Seeded graph matching for correlated erdös-rényi graphs. *J. Mach. Learn. Res.*, 15(1):3513–3540, January 2014. ISSN 1532-4435.

[49] Vince Lyzinski, Donniell E. Fishkind, Marcelo Fiori, Joshua T. Vogelstein, Carey E. Priebe, and Guillermo Sapiro. Graph matching: Relax at your own risk. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):60–73, 2016. doi: 10.1109/TPAMI.2015.2424894.

[50] Riccardo Marin, Simone Melzi, Emanuele Rodolà, and Umberto Castellani. Farm: Functional automatic registration method for 3d human bodies. *Comput. Graph. Forum*, 39(1):160–173, 2020. URL `https://doi.org/10.1111/cgf.13751`.

[51] Simone Melzi, Jing Ren, Emanuele Rodolà, Abhishek Sharma, Peter Wonka, and Maks Ovsjanikov. Zoomout: Spectral upsampling for efficient shape correspondence. *ACM Transactions on Graphics (TOG)*, 38(6):155, 2019.

[52] Maks Ovsjanikov, Jian Sun, and Leo Guibas. Global intrinsic symmetries of shapes. *Comp. Graph. Forum*, 27(5):1341–1348, 2008.

[53] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: A flexible representation of maps between shapes. *ACM Trans. Graph.*, 31(4), July 2012. ISSN 0730-0301. doi: 10.1145/2185520.2185526.

[54] Gautam Pai, Jing Ren, Simone Melzi, Peter Wonka, and Maks Ovsjanikov. Fast sinkhorn filters: Using matrix scaling for non-rigid shape correspondence with functional maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 384–393, June 2021.

[55] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *ACM Trans. Graph.*, 37(6), dec 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275040.

[56] Jing Ren, Mikhail Panine, Peter Wonka, and Maks Ovsjanikov. Structured regularization of functional map computations. *Comput. Graph. Forum*, 38(5):39–53, 2019. doi: 10.1111/cgf.13788. URL `https://doi.org/10.1111/cgf.13788`.

[57] E. Rodolí, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers. Partial functional correspondence. *Comput. Graph. Forum*, 36(1):222–236, January 2017. ISSN 0167-7055. doi: 10.1111/cgf.12797.

[58] Emanuele Rodolà, Samuel Bulò, Thomas Windheuser, Matthias Vestner, and Daniel Cremers. Dense non-rigid shape correspondence using random forests. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4177–4184, 2014. doi: 10.1109/CVPR.2014.532.

[59] Yusuf Sahillioğlu and Yücel Yemez. Minimum-distortion isometric shape correspondence using em algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11): 2203–2215, 2012. doi: 10.1109/TPAMI.2012.26.

[60] Aliaksei Sandryhaila and José M. F. Moura. Discrete signal processing on graphs. *IEEE Transactions on Signal Processing*, 61(7):1644–1656, 2013. doi: 10.1109/TSP.2013.2238935.

[61] Gil Shamai and Ron Kimmel. Geodesic distance descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[62] Tianjia Shao, Wilmot Li, Kun Zhou, Weiwei Xu, Baining Guo, and Niloy J. Mitra. Interpreting concept sketches. *ACM Trans. Graph.*, 32(4), July 2013. ISSN 0730-0301. doi: 10.1145/2461912.2462003.

[63] Roded Sharan and Trey Ideker. T. modeling cellular machinery through biological network comparison. nat. biotechnol. 24, 427-433. *Nature biotechnology*, 24:427–33, 05 2006. doi: 10.1038/nbt1196.

[64] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013. doi: 10.1109/MSP.2012.2235192.

[65] Rohit Singh, Jinbo Xu, and Bonnie Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences of the United States of America*, 105:12763–8, 10 2008. doi: 10.1073/pnas.0806627105.

[66] Justin Solomon, Andy Nguyen, Adrian Butscher, Mirela Ben-Chen, and Leonidas Guibas. Soft maps between surfaces. *Comput. Graph. Forum*, 31(5):1617–1626, August 2012. ISSN 0167-7055. doi: 10.1111/j.1467-8659.2012.03167.x.

[67] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *European conference on computer vision*, pages 356–369. Springer, 2010.

[68] Huynh Thanh Trung, Tong Van Vinh, Nguyen Thanh Tam, Hongzhi Yin, Matthias Weidlich, and Nguyen Quoc Viet Hung. Adaptive network alignment with unsupervised and multi-order convolutional networks. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 85–96, 2020. doi: 10.1109/ICDE48307.2020.00015.

[69] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988. doi: 10.1109/34.6778.

[70] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. accepted as poster.

[71] Mario Vento and Pasquale Foggia. Graph matching techniques for computer vision. *Graph-Based Methods in Computer Vision: Developments and Applications*, 1:1–41, 01 2012. doi: 10.4018/978-1-4666-1891-6.ch001.

[72] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: a deep attentional embedding approach. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 3670–3676, 2019. doi: 10.24963/ijcai.2019/509.

[73] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Learning combinatorial embedding networks for deep graph matching. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019. doi: 10.1109/iccv.2019.00315.

[74] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 349–357, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1032.

[75] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6861–6871. PMLR, 2019.

[76] Rui Xiang, Rongjie Lai, and Hongkai Zhao. Efficient and robust shape correspondence via sparsity-enforced quadratic assignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9513–9522, 2020.

[77] Rui Xiang, Rongjie Lai, and Hongkai Zhao. A dual iterative refinement method for non-rigid shape matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15930–15939, June 2021.

[78] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[79] Lei Xu and Irwin King. A pca approach for fast retrieval of structural patterns in attributed graphs. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 31:812–7, 02 2001. doi: 10.1109/3477.956043.

[80] Lyudmila Yartseva and Matthias Grossglauser. On the performance of percolation graph matching. In *COSN '13*, 2013.

[81] LIREN YU, Jiaming Xu, and Xiaojun Lin. Graph matching with partially-correct seeds. *J. Mach. Learn. Res.*, 22:280:1–280:54, 2021.

[82] Andrei Zanfir and Cristian Sminchisescu. Deep learning of graph matching. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2684–2693, 2018. doi: 10.1109/CVPR.2018.00284.

[83] Jiawei Zhang and Philip S. Yu. Multiple anonymized social networks alignment. In *2015 IEEE International Conference on Data Mining*, pages 599–608, 2015. doi: 10.1109/ICDM.2015.114.

[84] Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. Attributed graph clustering via adaptive graph convolution. In *IJCAI*, 2019.

[85] Zhen Zhang and Wee Sun Lee. Deep graphical feature learning for the feature matching problem. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5086–5095, 2019. doi: 10.1109/ICCV.2019.00519.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes]
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] see Sections 1, 3 4, 5 and Appendix D
   (b) Did you describe the limitations of your work? [Yes] see Section 5.7
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] see Section 7
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] see Sections 3, 2.2 and Appendix C
   (b) Did you include complete proofs of all theoretical results? [Yes] see Sections listed in Question 2 (a) above.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] see code in supplementary material
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see Section 5.5 amd Appendix D
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] see Section 5.1
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] see Section 5.5 and Appendix D.4
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] see code in supplementary material
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] see Section 5.5 and Appendix D.4

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Exact Shape Correspondence via 2D Graph Convolution.

Appendices:

A Related work

A.1 point-wise Alignment

Given a set of point-wise descriptors $\mathbf{X}_A \in \mathbb{R}^{n_1 \times d}$ and $\mathbf{X}_B \in \mathbb{R}^{n_2 \times d}$ for two 3D shapes $\mathcal{M}_1$ and $\mathcal{M}_2$, such as the ones based on (1) the Laplacian Beltrami operator (LBO) e.g., Heat Kernel Signatures (HKS) [11], Wave Kernel Signatures (WKS) [6], Global point Signatures (GPS) [52], which all use a truncated basis of the eigen-functions (eigen-vectors) of the LBO rather than using the full basis, or given point-wise descriptors based on (2) the Geodesic distances e.g., Geodesic distance descriptor (GDD) [61], or (3) others based on signatures of histograms such as SHOT [67], the *point-wise alignment* problem can be formulated as:

$$\min_{\mathbf{P}} ||\mathbf{P}\mathbf{X}_A - \mathbf{X}_B||_F^2 = \max_{\mathbf{P}} \mathbf{Tr}(\mathbf{P}^T \mathbf{X}_A \mathbf{X}_B^T), \tag{12}$$

As observed by Lähner et al. [39], there are two main problems with this approach. Which are: (a) despite the variety of point-wise shape descriptors proposed, most assume near-isometry (i.e., assume no noise), and (b) shapes usually have intrinsic symmetries (e.g., humans left side and right side are almost perfect copies of each other) which is not easily captured by point-wise descriptors. The later problem will result in serious discontinuities in the map as some points will be mapped to their desired location, but others to their symmetric counterparts e.g., the left eye and right eye of a human face both being mapped to the right eye of the other face. As such For 2D-GEM we not only use poit-wise descriptors, but combine them with pair-wise descriptors via the 2D-graph convolution.

A.2 pair-wise Alignment.

Given a set of pair-wise descriptors $\mathbf{A} \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{B} \in \mathbb{R}^{n_2 \times n_2}$ for two 3D shapes $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively, such as their; Laplacians, adjacency matrices, geodesic distance matrices, mass matrices, or stiffness matrices [76, 39], the alignment problem is given as the quadratic assignment problem (QAP) [21]:

$$\min_{\mathbf{P}} ||\mathbf{P}\mathbf{A}\mathbf{P}^T - \mathbf{B}||_F^2 = \max_{\mathbf{P}} \mathbf{Tr}(\mathbf{P}^T \mathbf{A}\mathbf{P}\mathbf{B}), \tag{13}$$

The QAP problem is computationally intractable. Thus, for 2D-GEM we employ the 2D-graph convolution which helps us reduce the alignment problem to a Linear Assignment Problem (LAP). It is notable to mention however that several relaxations have been proposed for the QAP problem (some of which are grouped as spectral methods and Quadratic Programming methods below).

Spectral Methods. These methods replace the requirement for permutation matrices with a single constraint on the Frobenious norm of the matrices to obtain a maximal eigenvalue problem [62, 40, 47]. They are used both for testing graph isomorphism [34, 7] and for graph matching. They usually reduce (13) into a Linear Assignment Problem (LAP) of the form $\max_{\mathbf{P}} \mathbf{Tr}(\mathbf{P}\mathbf{C})$, where we try to find the $\mathbf{P}$ permutation matrix that achieves the best alignment of the eigenvectors, where $\mathbf{C}$ is the similarity (cost) matrix of the eigen-vectors. Given two graphs $\mathcal{A}$ and $\mathcal{B}$ with adjacencies $\mathbf{A} = \sum_i \lambda_i \mathbf{u}_i \mathbf{u}_i^T$ and $\mathbf{B} = \sum_j \mu_j \mathbf{v}_j \mathbf{v}_j^T$, where $\lambda_i, \mathbf{u}_i$ are the eigenvalues and eigenvectors of $\mathbf{A}$ and $\mu_j, \mathbf{v}_j$ those of $\mathbf{B}$, Umeyama [69] defined his cost function as $\mathbf{C} = \sum_{i,j} s_i \mathbf{u}_i \mathbf{v}_i^T$, for some appropriately chosen signs $s_i \in \{\pm 1\}$ (similar to Xu and King [79]). The motivation being that, this is the solution to the orthogonal relaxation of (13), where the feasible set is relaxed to the set the orthogonal matrices [21]). Furthermore Umeyama [69] suggested the construction $\mathbf{C} = \sum_{i,j} |\mathbf{u}_i||\mathbf{v}_i|^T$, where $|\mathbf{u}_i|$ is the element wise absolute value of $\mathbf{u}_i$. Kazemi et al. [29] and Feizi et al. [19] proposed other low rank cost functions, the simplest of which is $\mathbf{C} = \mathbf{u}_1 \mathbf{v}_1^T$. Fan et al. [18] proposed GRAMPA, suggesting that using a combination of all eigenvectors rather than only those where $i = j$ was significantly more

resilient to noise and proposed $\mathbf{C} = \sum_{i,j} w(\lambda_i, \mu_j)\mathbf{u}_i\mathbf{u}_i^T\mathbf{J}\mathbf{v}_j\mathbf{v}_j^T$, where $w(\lambda, u)$ is a monotonically decreasing function such as the Cauchy spectral weight $w(\lambda, \mu) = \frac{1}{(\lambda-\mu)^2+\gamma^2}$, γ being a bandwidth parameter. Two other notable Spectral methods related to 2D-GEM are Fast Sinkhorn Filters [54] and Structured Regularization of Functional Map Computations [56]. [54] provides a unifying framework for spectral methods. Following the Meta Algorithm from [54], 2D-GEM can equally be seen as a spectral alignment regularised algorithm. [54] proposed the Sinkhorn filter instead of the KNN and Auction (and GMWM) matching. It is worth considering to incorporate the sinkhorn filter into 2D-GEM. Concerning the differences, besides the difference in the motivation for Fast Sinhorn Filters (Regularized optimal transport) and 2D-GEM (spectral graph theory), the main differences between them are first the fact that the cost matrix proposed in Fast Sinkhorn Filters is a distance matrix (euclidean) between the eigen-vectors, while that in 2D-GEM is based on the spectrum of the graphs of the corresponding shapes. Second, rather than using single eigen-vectors on each shape, 2D-GEM uses pairs of eigen-vectors for its cost matrix. Finally, the regularizers used for 2D-GEM enforce both neighbourhoods preservation (via the 2-Hop algorithm) and geodesic distortion preservation (via the LMD), while the regularizer in fast Sinkhorn Filters is mainly to make the problem strictly convex which gives the interpretation of a regularized transport plan. [56] pointed out the fact that only penalizing the failure of the unknown map to commute with the Laplace-Beltrami operator operators on the source and target shapes may fail in some cases. To address this problem, [56] further proposed additional regularizers to update the functional map (namely, a descriptor-commutative regularizing term, an orientation-preserving term, and a resolvent term). 2D-GEM may be used to attempt to bridge the gap between the popular Functional Map and the 2D-Graph convolution. In this regard, 2D-GEM can also be interpreted as a regularised spectral alignment method to update the map. Thus, it is related to [56] with the difference that the regularizers in our case are neighborhood preservation (enforced via the 2-Hop) and distortion preservation (enforced via the LMD) in addition to the fact that the map in our case is computed in the spectrum of the graphs associated with the shape's triangulation rather than in the traditional shape LBO's spectrum.

Quadratic Programming (QP), Doubly Stochastic (DS) and Semi Definite Programing (SDP) Methods. Here we have other relaxations of (13) e.g., denoting the set of permutation matrices in $\mathbb{R}^{n\times n}$ by $\mathbf{P}$, notice that (13) can be reformulated as:

$$\min_{\mathbf{P}\in\Pi} ||\mathbf{AP} - \mathbf{PB}||_F^2 \tag{14}$$

Relaxing the set of permutation matrices Π to its convex hull (the Birkoff polytope of doubly stochastic matrices $\mathcal{P}_B$), leads to the doubly stochastic relaxation [66, 1]. Other methods here [16] add different regularization terms to (14) or use different relaxations like the Linear Programming (LP) [3] or the Semi Definite Programming (SDP) [30]. GRAMPA can also be placed in this category as its cost matrix is a solution to the regularized QP.

Neural Network Methods. Various Graph neural networks (GNNs) have been proposed for various graph related tasks [32, 33, 24, 70, 68, 42, 72, 84, 75, 78, 72, 22]. It was recently shown that these GNNs can work as the Weisfeiler-Lehman [78, 15]. It was shown that for the graph isomorphism task amongst these GNNs, those that update a node's representation as a sum of itself and its neighbours features work better than those which use other aggregation techniques. As such, many of these GNNs have been used for the graph matching and graph similarity tasks [82, 25, 85, 73, 74, 8, 9, 44]. Many of these methods leverage the graph convolution for the shape matching task as well. However, they perform the convolution on the features of each individual 3D shape as a way to combine the pair-wise and point-wise descriptors of each of the shapes and then feed these combined features into a learning algorithm for supervised or unsupervised alignment using a suitable loss function [20]. Our method on the other hand is unsupervised and does not use neural nets, and our proposed filter is derived from our new definition of noise in the context graph matching and can be regarded as a 2D graph convolution over the point-wise descriptor similarity space or over the initialized correspondence.

Other Methods. Other related methods for the graph alignment task include; Symmetric-Non-Negative-Matrix-Factorization methods for graph and bipartite graph matching [13, 37]. We equally have the popular Graduated Assignment [23] in this category, as well as the ISORANK algorithm [65] which is based on the adjacency matrix of product graphs. Kollias et al. [36] proposed Network Similarity Decomposition (NSD) to approximate ISORANK by using decomposition techniques, thus avoiding the quadratic size of the product graph. In "noisy" settings where the graphs are not exactly isomorphic, there have been many proposed algorithms [80, 48, 49, 14] besides GRAMPA.

A.3 Joint Alignment (Both)

Using some of the aforementioned pair-wise descriptors $\mathbf{A} \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{B} \in \mathbb{R}^{n_2 \times n_2}$, and point-wise descriptors $\mathbf{X}_A \in \mathbb{R}^{n_1 \times d}$ and $\mathbf{X}_B \in \mathbb{R}^{n_2 \times d}$ for two 3D shapes $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively , these methods are either of the form:

$$\min_{\mathbf{P} \in \Pi} ||\mathbf{P}\mathbf{A}\mathbf{P}^T - \mathbf{B}||_F^2 + \alpha ||\mathbf{P}\mathbf{X}_A - \mathbf{X}_B||_F^2, \tag{15}$$

such as SEQA, where the parameter α acts as a trade of parameter between the *pair-wise alignment* and the *point-wise alignment*. Or of the form:

$$\max_{\mathbf{P} \in \Pi} \mathbf{Tr}(\mathbf{P}^T \mathbf{A}\mathbf{P}\mathbf{B}) + \alpha \mathbf{Tr}(\mathbf{P}^T \mathbf{X}_A \mathbf{X}_B^T), \tag{16}$$

such as Kernel-Match. The first term in both (15) and (16) will normally lead to the QAP which is computationally intractable, but to avoid this, these two methods relax $\mathbf{P}$ to be in the set of doubly stochastic matrices $\mathcal{P}_B$ rather the set of permutation matrices Π. One can observe that methods in this category try to find a trade-off between the contribution of the point-wise descriptors and those of the pair-wise descriptors to the assignment problem. Moreover these methods have traditionally been shown [39, 76] to outperform *pair-wise alignment* and *point-wise alignment* methods. Hence, inspired by the performance of these methods, we proposed to leverage the 2D graph convolution instead for better performance at a relatively good time complexity as it equally leads to an LAP rather than the QAP.

Other methods in this category usually initialize the solution $\mathbf{P}$ from point-wise alignment by solving (12), and then iteratively update the solution using a relaxation of pair-wise alignment e.g., using a truncated basis space via the notion of Functional Maps [53]. These methods include; ICP [4], ZoomOut [51], and others [55, 61].

For nearly-isometric shapes, the approach used by most of these methods (i.e., first finding an initial correspondence, and then iteratively improving this initial correspondence using all points and eigen-vectors corresponding to lower frequencies (eigen-values)), leads to an erroneous final correspondence due to accumulation of errors in the updates given that the initial correspondence is usually not exact [77]. Moreover, these methods are usually not very noise robust as they usually assume near-isometry between the shapes, thus, they equally do not perform well on the task of exact matching on non-isometric shapes. Hence, for 2D-GEM we propose a 2D Graph convolution-based framework that is robust to noise on non-isometric shapes, and can incorporate more tight constraints (such as preservation of geodesic distances) to reduce the error accumulation in the correspondence update on near-isometric shapes.

B GRAMPA for Pair-wise Alignment

In this section we review GRAMPA [18] for 3D shape matching via *pair-wise alignment*. This algorithm consists of two parts; (1) Formulation/Initialization of the permutation matrix, and (2) iterative improvement of the initialized correspondence.

B.1 Formulation

In graph theory, an isomorphism of graphs $\mathcal{A}$ and $\mathcal{B}$ is a bijection between the vertex sets of $\mathcal{A}$ and $\mathcal{B}$ i.e., $f\colon V(\mathcal{A}) \to V(\mathcal{B})$, such that any two vertices u and v of $\mathcal{A}$ are adjacent in $\mathcal{A}$ if and only if $f(u)$ and $f(v)$ are adjacent in $\mathcal{B}$. If an isomorphism exists between two graphs, then the graphs are called isomorphic and denoted as $\mathcal{A} \simeq \mathcal{B}$. From linear algebra, two graphs $\mathcal{A}$ and $\mathcal{B}$ are isomorphic iff the adjacency matrix of one can be expressed as the permutation of the rows and columns of the other i.e.,

$$\mathbf{A} = \mathbf{P}^T \mathbf{B} \mathbf{P}, \tag{17}$$

where , and $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ are the adjacency matrices of $\mathcal{A}$ and $\mathcal{B}$ respectively. As such, the graph matching problem [41, 20, 73, 18, 12, 17] is the attempt at finding the permutation matrix $\mathbf{P}$ such that (17) holds.

This problem has several applications which include but are not limited to action identification in computer vision [71], protein network matching in bioinformatics[63, 65], comparing molecules in

cheminformatics [35], and linking user accounts (de-anonymization) in social network analysis [83]. GRAMPA was proposed to address this problem in a noisy setting, where both graphs may not be exactly isomorphic.

For 3D shape matching, graph matching can be seen as a *pair-wise alignment* problem in which the set of pair-wise descriptors are the adjacency matrices. Using the adjacency matrices $\mathbf{A} \in \mathbb{R}^{n_1 \times n_1}$ and $\mathbf{B} \in \mathbb{R}^{n_2 \times n_2}$ for two 3D shapes $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively, the alignment problem is given as the Quadratic Assignment Problem (QAP) [21] as in equation (13).

Several relaxations of this problem have been proposed [62, 40, 47, 69, 29, 19, 66, 1, 16, 3, 30]. Amongst the Spectral Relaxations, GRAMPA has been proven to achieve the SOTA results, being tobust to high noise levels[2]. GRAMPA's relaxed alignment is $\max_{\mathbf{P}} \mathbf{Tr}(\mathbf{P}^T\mathbf{C})$, where $\mathbf{C}$ is their cost matrix given as:

$$\mathbf{C} = \sum_{i,j} w(\lambda_i, \mu_j)\mathbf{u}_i\mathbf{u}_i^T\mathbf{J}_{A,B}\mathbf{v}_j\mathbf{v}_j^T, \tag{18}$$

$\mathbf{Tr}(\cdot)$ is the matrix Trace operator, $w(\lambda, \mu) = \frac{1}{(\lambda-\mu)^2+\gamma^2}$, and γ the noise control parameter. This cost matrix $\mathbf{C}$ in (18) was shown by Fan et al. [18] to be the solution $\mathbf{X}$ to the regularized relaxed QAP:

$$\min_{X \in \mathbb{R}^{n \times n}} \frac{1}{2}||\mathbf{AX} - \mathbf{XB}||_F^2 + \frac{\gamma^2}{2} + ||\mathbf{X}||_F^2 - \mathbf{1X1}, \tag{19}$$

first relaxing the set of permutations Π to its convex hull (the Birkhoff polytope of doubly stochastic matrices), and further relaxing to only be required to satisfy the constraint $\mathbf{1X1} = n$. The proof being tedious, we refer the reader to their paper [18].

In the case where $n_1 \neq n_2$, the permutation matrix $\mathbf{P} \in \mathbb{R}^{n_2 \times n_1}$ is adjusted to be a rectangular assignment matrix. This problem has been widely addressed not only on 3D shapes, but also generally on graphs.

B.2 2-Hop Iterative improvement

Once the permutation matrix $\mathbf{P}$ is obtained, GRAMPA proposed to use the iterative cleanup procedure: $\mathbf{P}_{t+1} = \arg\max_{\mathbf{P} \in \Pi} \mathbf{Tr}(\mathbf{P}^T\mathbf{A}\mathbf{P}_t\mathbf{B})$. More recently it has been suggested to use j-hop neighbourhood adjacency matrices $\mathbf{A}_j$ and $\mathbf{B}_j$ for the iterative improvement instead [81]. Specifically, YU et al. [81] suggested to use the 2-hop neighborhood and gave performance guarantees. The 2-hop adjacency matrices $\mathbf{A}_2$ and $\mathbf{B}_2$ (using the Delauney triangulations), are such that $[\mathbf{A}_2]_{i,j} = 1$ and $[\mathbf{B}_2]_{i,j} = 1$ if i and j are 2-hop neighbors, and 0 otherwise. Using these, the iterative improvement becomes $\mathbf{P}_{t+1} = \arg\max_{\mathbf{P} \in \Pi} \mathbf{Tr}(\mathbf{P}^T\mathbf{A}_2\mathbf{P}_t\mathbf{B}_2)$.

B.3 Relationship Between 2D-GEM and GRAMPA

Looking at GRAMPA's cost function in equation (18), one can see that using the adjacency instead of the Laplacian, as well as using no features, 2D-GEM as in equation (7) becomes exactly GRAMPA with a similar but different filter $w(\lambda_i, \mu_j)$. As such, GRAMPA can also be seen as filtering [60] the initialized indistinguishable 2D signals of graphs using the adjacency basis and a graph-matching-suitable-filter (i.e., $w(\lambda, \mu)$), and then iteratively updating the correspondence i.e., GRAMPA is also a filter (de-noise), then initialize, then update algorithm. However, 2D-GEM as a 2D graph convolution allows us to utilise features (such as the *point-wise* descriptors in the context of shape matching), which causes 2D-GEM to achieve better performance than GRAMPA when these features are informative. Besides comparison on the 3D shape matching task, we equally do more empirical comparisons between 2D-GEM and GRAMPA on random graphs following the Erdos Regny model (see Appendix D). These experiments show that indeed when we have features that are informative 2D-GEM may not only outperform GRAMPA, but also the QP (i.e., the full Quadratic Assignment Problem with full doubly stochastic constraints [18]) depending on how informative the features are.

[2] we refer the reader to their paper [18].

C On the In-variance of the Graph Convolution to Permutation

In this section we want to review graph isomorphism from linear algebra and revisit the in-variance of graph convolution to permutation.

C.1 Review of Isomorphism from Linear algebra

From linear algebra, to graphs $\mathcal{A}$ and $\mathcal{B}$ are isomorphic iff the adjacency matrix of one can be expressed as the permutation of the rows and columns of the other i.e.,

$$\mathbf{A} = \mathbf{P}^T\mathbf{B}\mathbf{P}, \tag{20}$$

where $\mathbf{P} \in \mathbb{R}^{n\times n}$ is a permutation matrix, and $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times n}$ are the adjacency matrices of $\mathcal{A}$ and $\mathcal{B}$. This implies that the diagonal degree matrix $\mathbf{D}_A$ of $\mathbf{A}$ is also a permutation of the diagonal degree matrix $\mathbf{D}_B$ of matrix $\mathbf{B}$:

$$\mathbf{d}_A = \mathbf{A}\mathbf{1} = \mathbf{P}^T\mathbf{B}\mathbf{P}\mathbf{1} = \mathbf{P}^T\mathbf{d}_B, \tag{21}$$

where $\mathbf{d}_A, \mathbf{d}_B \in \mathbb{R}^n$ are the vectors of degree, $\mathbf{1} \in \mathbb{R}^n$ is the all ones vector, $\mathbf{P}\mathbf{1} = \mathbf{1}$. This can be written in matrix form as:

$$\mathbf{D}_A = \mathbf{P}^T\mathbf{D}_B\mathbf{P}, \tag{22}$$

where the extra $\mathbf{P}$ on the left of $\mathbf{D}_B$ ensures that the permutation keeps the node degree with the corresponding node index of the rows and columns of $\mathbf{D}_B$. One can observe that a linear combination of (20) and (22) will imply that isomorphism also implies that the Laplacian matrix of $\mathcal{A}$ is also a permutation of that of $\mathcal{B}$, i.e., taking (22) - (20), we have:

$$\begin{aligned}
\mathbf{L}_A = \mathbf{D}_A - \mathbf{A} &= \mathbf{P}^T\mathbf{D}_B\mathbf{P} - \mathbf{P}^T\mathbf{B}\mathbf{P} \\
&= \mathbf{P}^T(\mathbf{D}_B - \mathbf{B})\mathbf{P} = \mathbf{P}^T\mathbf{L}_B\mathbf{P} \\
&\rightarrow \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T = \mathbf{P}^T\mathbf{V}\mathbf{M}\mathbf{V}^T\mathbf{P} \\
&\rightarrow \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^T = \mathbf{P}^T\mathbf{V}h(\mathbf{M})\mathbf{V}^T\mathbf{P},
\end{aligned} \tag{23}$$

because $\mathbf{\Lambda} = \mathbf{M}$ for isomorphic graphs, we also have $g(\mathbf{\Lambda}) = h(\mathbf{M})$ where $\mathbf{L}_A \in \mathbb{R}^{n\times n}$ and $\mathbf{L}_B \in \mathbb{R}^{n\times n}$ are the Laplacians of $\mathcal{A}$ and $\mathcal{B}$ respectively, $\mathbf{\Lambda}, \mathbf{M} \in \mathbb{R}^{n\times n}$ the eigen-value diagonal matrices, $g(\mathbf{\Lambda})$, $h(\mathbf{M})$ functions of the eigen-values (where g and h are the same function e.g., $g(\lambda) = 1 - \alpha\lambda$, and $h(\mu) = 1 - \alpha\mu$), and $\mathbf{U}, \mathbf{V}$ and the eigen-vectors of the Laplacians $\mathbf{L}_A$ and $\mathbf{L}_B$ respectively.

Moreover, if $\mathcal{A}$ and $\mathcal{B}$ are associated with node attributes $\mathbf{X}_A \in \mathbb{R}^{n\times d}$ and $\mathbf{X}_B \in \mathbb{R}^{n\times d}$, one can also observe that:

$$\mathbf{X}_A = \mathbf{P}^T\mathbf{X}_B, \tag{24}$$

i.e., there exists a permutation of the node attribute matrix of $\mathcal{B}$ that reorders it into the node attribute matrix of $\mathcal{A}$. However, one should note that here isomorphism implies equation (24) holds, but equation (24) holding does not necessarily imply isomorphism. So, here we can be seen as using some form of relaxation. Notice also that this relaxation is reasonable (and made by other baselines [39]) as in real life meaningful features often lead to some level of information on the permutation matrix.

Putting (23) and (24) together, we have the in-variance of the graph convolution to permutation [20], i.e., for any convolution function h that allows the diagonalization $h(\mathbf{L}_B) = \mathbf{V}h(\mathbf{M})\mathbf{V}^T$ we have that:

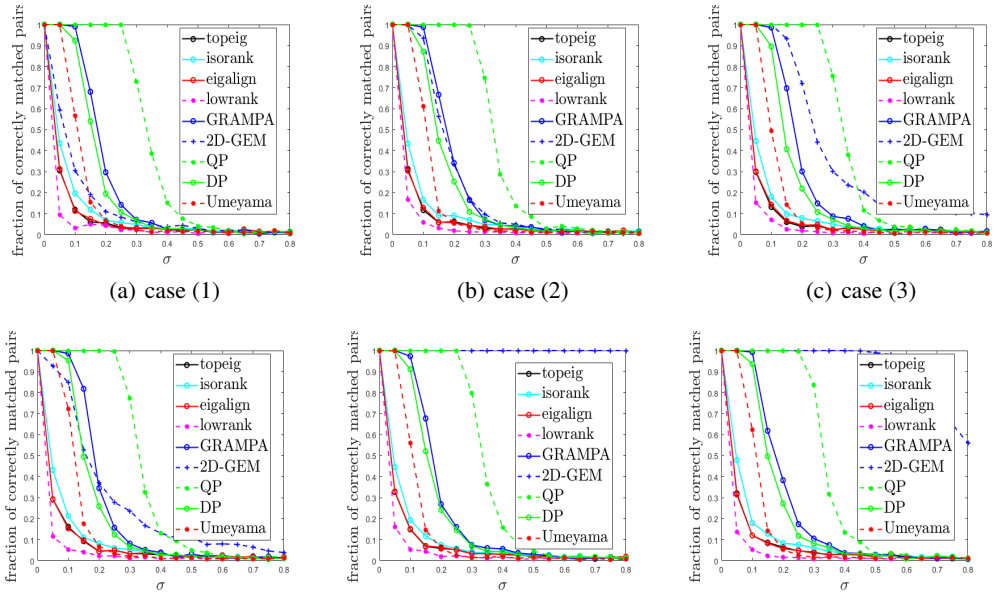
$$\mathbf{P}^T\mathbf{V}h(\mathbf{M})\mathbf{V}^T\mathbf{P}\mathbf{P}^T\mathbf{X}_B = \mathbf{P}^T\mathbf{V}h(\mathbf{M})\mathbf{V}^T\mathbf{X}_B \tag{25}$$

where $\mathbf{P}\mathbf{P}^T = \mathbf{P}^T\mathbf{P} = \mathbf{I}$ for permutation matrices $\mathbf{P}$. As such we have that for isomorphism,

$$\begin{aligned}
\mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^T\mathbf{X}_A &= \mathbf{P}^T\mathbf{V}h(\mathbf{M})\mathbf{V}^T\mathbf{X}_B \\
&\rightarrow \mathbf{P}\mathbb{M}_A\mathbf{X}_A = \mathbb{M}_B\mathbf{X}_B,
\end{aligned} \tag{26}$$

where $\mathbb{M}_A = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^T = g(\mathbf{L}_A)$ is the convolution function for graph $\mathcal{A}$, and likewise $\mathbb{M}_B = \mathbf{V}h(\mathbf{M})\mathbf{V}^T = h(\mathbf{L}_B)$ is the convolution function for graph $\mathcal{B}$. As such the graph matching problem can be reformulated using the convolution as:

$$\min_{\mathbf{P}} ||\mathbf{P}\mathbb{M}_A\mathbf{X}_A - \mathbb{M}_B\mathbf{X}_B||_F^2, \tag{27}$$



(a) case (1) (b) case (2) (c) case (3)

(d) case (4) (e) case (5) (f) case (6)

Figure 6: Plots on Erdos Regny random graphs, using $n = 100$ and $p = 0.5$. All plots follow the same legend as case (1), Cases (1)-(2) use undistinguishable features i.e., the all ones as in equation (5)), cases (3)-(4) use features such that 10% of the nodes are distinguishable, and cases (5)-(6) such that 100% of the nodes are distinguishable. Cases (1)-(3), and (5) use all eigen-values, while cases (4) and (6) use 20 eigen-values as in equation (8). All cases use $e = 10$ except case (1) where $e = 1$.Nodes with distinguishable features are chosen at random.

D Validation of 2D Graph Convolution for Graph Matching on Erdos Regny Random Graphs

In this section we conduct experiments to validate the effectiveness of utilizing features available via 2D graph convolution for graph matching. We compare the 2D graph convolution (referred to as 2D-GEM also in this Section, where we do not use any LMD nor 2-hop refinement, but just the cost in equation (7) or equation (8) and then the normal rounding procedure "matchpairs" provided by Matlab) with GRAMPA and few other methods (which all cannot utilize features). We make use of the $\mathcal{G}(n, q)$ Erdos Regny random graphs model, where n determines the number of nodes and q the number of edges in the graph.

D.1 Datasets

Given a parent Erdos Regny random graph $\mathcal{G}(n, q)$ with adjacency $\mathbf{G}$, we first sample a graph from it which has the same number of nodes n but deleting edges (each independently) from it with probability $1 - s$, where $s \in [0, 1]$ to obtain a source graph with adjacency $\mathbf{A} \sim \mathcal{G}(n, p)$, where $q \triangleq ps$. We then define another graph (target graph) with adjacency $\mathbf{B} \sim \mathcal{G}(n, p)$ as:

$$\mathbf{B}_{i,j} \sim \begin{cases} \text{Bern}(s) & \text{if } \mathbf{A}_{ij} = 1 \\ \text{Bern}\left(\frac{p(1-s)}{1-p}\right) & \text{if } \mathbf{A}_{ij} = 0 \end{cases}, \tag{28}$$

where $\text{Bern}(s)$ denotes a Bernoulli distribution with mean s. Note that $\sigma^2 = 1 - s$ can be seen as the square of the noise level determining the number of edges that differ between $\mathbf{A}$ and $\mathbf{B}$. Suppose we observe a pair of graphs with adjacency matrices $\mathbf{A}$ and $\mathbf{B}' = \mathbf{P}^T\mathbf{B}\mathbf{P}$, where $\mathbf{P}$ is the permutation matrix. We then wish to recover $\mathbf{P}$.

D.2 Evaluation Metrics

For these experiments on random Erdos Regny graphs, we visualise the $\%$ of correct correspondences against the noise level σ.

D.3 Baselines

We compare 2D-GEM to GRAMPA [18], and others such as: Umeyama [69], Degree Profile (DP) [14], ISORANK [28], QP (where we follow [18] and use the relaxation of the permutation matrix in the Quadratic Assignment Problem to a doubly stochastic matrix), EigenAlign [19], LowrankAlign [19] and TopEigenVec [18].

D.4 Parameter Settings

We use graphs with $n = 100$ and set $p = 0.5$ and σ starting from 0, and using a step size of 0.05 up to 0.8. We repeat the experiments 10 times each and each time we generate features for the graphs we generate features $\mathbf{X}_A$ for the graph with adjacency $\mathbf{A}$ and use $\mathbf{X}_B = \mathbf{P}^T\mathbf{X}_A$ as features for $\mathbf{B}$.

We conduct four different experiments and run each 10 times and plot the averages. The details for each setting for 2D-GEM are as follows (the other baselines are set following Fan et al. [18] who kindly provided their source code):

- case (1): for figure (6(a)) we use indistinguishable features $\mathbf{J}_{A,B}$, and use all the eigen-values for 2D-GEM as in equation (5) with $e = 1$,
- case (2): for figure (6(b)) we use indistinguishable features $\mathbf{J}_{A,B}$, all eigen-values and use $e = 10$,
- case (3): for figure figure (6(c)) we use features such that 10% of randomly selected nodes are distinguishable (i.e., 10% of nodes in both graphs can be matched using these features) and use all the eigen-values for 2D-GEM as in equation (7), and $e = 10$
- case (4): for figure figure (6(d)) we use features such that 10% of randomly selected nodes are distinguishable and use 20 eigen-values (instead of $n = 100$ eigen-values) for 2D-GEM as in equation (8), and $e = 10$
- case (5): for figure figure (6(e)) we use features such that all nodes are distinguishable and use $e = 10$ and all eigen-values for 2D-GEM as in equation (7),
- case (6): for figure figure (6(f)) we use features such that all nodes are distinguishable and use $e = 10$ and 20 eigen-values (instead of $n = 100$ eigen-values) for 2D-GEM as in equation (8),

D.5 Performance Analysis

Cases (1) and (2) show that without features, 2D-GEM with $e = 1$ performs worse than GRAMPA and even other methods such as DP and Umeyama, while with $e = 10$, 2D-GEM performs on par with GRAMPA. This trend is in fact general and also occurs in other cases i.e., the performance of 2D-GEM increases as e increases. Cases (3)-(6) illustrate the fact that if we have features which are a little informative even as little as 10% (case (3)-(4)), 2D-GEM can outperform even GRAMPA, and if the features are fully informative (case (5)-(6)), 2D-GEM using all eigen-values (case (5)) can recover the exact permutation matrix no mater the noise level in the adjacency, while 2D-GEM using as little as 20 eigen-values also performs significantly well (case (6)).

These experiments demonstrate that using features in addition to the adjacency/Laplacian (i.e., *point-wise* descriptors together with *pair-wise* descriptors) will be more beneficial than using only either. Moreover, these experiments also show that it is non-detrimental to assume that the permutation of the adjacency and those of the features correspond, as even when the features a non-informative such as in cases (1)-(2) where we used the all ones, the performance of 2D-GEM was on par with GRAMPA, and the performance of 2D-GEM only got better as the features became more informative (cases (3)-(6)). Moreover, in real life, most graphs are associated with features and for 3D shapes, one could extract *point-wise* descriptors as well as *pair-wise* descriptors. A caution here is the case where the information contained in both do not agree with each other, though we did not test this case, it is obvious that in such a case the permutation matrix will be different.

Algorithm 3 : case (3)

1. **input** An initial correspondence $\mathbf{P}^0$, $\mathbf{U}_k$, $\mathbf{V}_k, \mathbf{\Lambda}_k, \mathbf{M}_k, e$, maximum iteration t for 2-hop neighborhood algorithm.

while $0 < i \leq iter$ **do**

2. Update $\mathbf{P}^i$ using 2-Hop with algorithm 2, where $t = 1$,
3. De-noise and update correspondence:
$\mathbf{P}^i = \arg\max_{\mathbf{P}} \mathbf{Tr}(\mathbf{PC})$,
where $\mathbf{C} = \sum_{i,j}^{k} w(\lambda_i, \mu_j)\mathbf{u}_i\mathbf{u}_i^T\mathbf{P}^i\mathbf{v}_j\mathbf{v}_j$,

end while

4. **return** $\mathbf{P}^t$.

E Ablation Studies on 2D-GEM

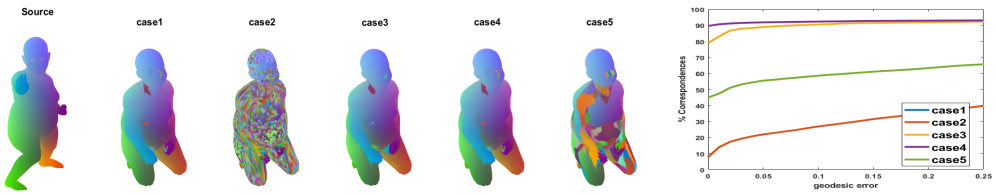
Here we conduct ablation studies on 2D-GEM to show the effectiveness of the overall framework over its individual parts.

E.1 Set-up

We compare 5 different cases (scenarios). For all cases, we use the same parameters as in Section 5.5, except that for all datasets we use 500 LBO eigen-vectors.

- case (1): we use 2D-GEM as it is,
- case (2): 2D-GEM without the 2-Hop improvement i.e., without step (2) in algorithm 1,
- case (3): 2D-GEM without the LMD. As in algorithm 3, with $iter = 60$,
- case (4): 2D-GEM with only the 2-Hop algorithm i.e., without the LMD nor the 2D graph convolution. As in algorithm 3 without step (3),
- case (5): 2D-GEM with only the 2D graph convolution i.e., without the LMD nor the 2-Hop algorithm. As in algorithm 3 without step (2).

E.2 Comparison

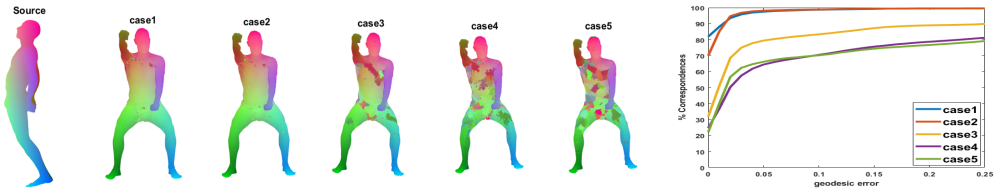


(a) Sample on TOPKIDS (b) Plot of sample on TOPKIDS

Figure 7: Ablation studies on non-isometric dataset (TOPKIDS). In Figure 7(b) above the plots of case 1 and 4 overlap)

Ablation on non-isometric shapes. One can observe from Figure 7(b) that cases (1) and (4) have exactly the same performance, since the plot of case (1) is shadowed by that of case (4). This suggests that for non-isometric shape matching, given a good initialization, the 2-Hop algorithm is as strong as 2D-GEM' framework (However, this is not the case on nearly-isometric shapes as seen in Figure 8(b) where it performs much worse than 2D-GEM). This equally shows that the 2D graph convolution provides such a good initialization. Moreover, it can be seen on this plot that the 2D graph convolution by itself (i.e., case (5)) performs moderately at geodesic 0 though it produces non-smooth maps. On the other hand, it can be observed that 2D-GEM with only the 2D convolution and the 2-Hop algorithm (i.e., case (3)) performs a little worse than 2D-GEM as a whole and than the 2-Hop algorithm by itself. It can also be seen on this plot that the LMD by itself (case (2) is really not suited for non-isometric shapes as for these shapes geodesic distances are not preserved.

Ablation on isometric shapes. Figure 8(b) shows that 2D-GEM as a whole (i.e., case (1)) performs best once again, and this time much better than the 2-Hop algorithm by itself (i.e., case (4)) since it



(a) Sample on SCAPE (b) Plot of sample on SCAPE

Figure 8: Ablation studies on isometric dataset (SCAPE).

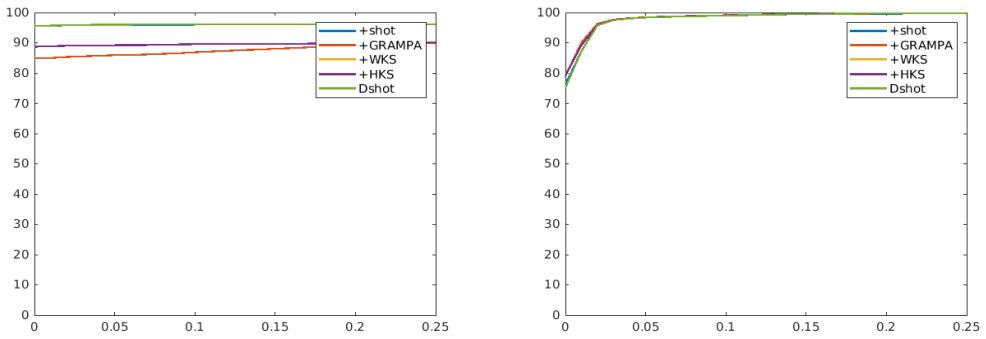
assumes conservation of geodesic distances (a constraint not accounted for by the 2-Hop algorithm). Moreover, the 2D graph convolution by itself (i.e., case (5)) is also seen to be on par and even outmatching the 2-Hop algorithm, while being outmatched by the combination of both (i.e., case (3)). Finally, it can equally be seen that the LMD (i.e., case (2)) is quite good for isometric shapes as it takes into account the preservation of geometric properties such as geodesic distances. However, the LMD (i.e., case (2)) is still outmatched in the exact matching task by 2D-GEM (i.e., case (1)) as can be seen on this figure at geodesic distance 0.

E.3 Ablation Conclusion

Overall from the ablation ablation studies, one can see that 2D-GEM helps one with the flexibility of incorporating stronger constraints such as preservation of geodesic distances or relaxing them, depending on the properties of the shapes at hand. Moreover, one can also see that as on random graphs (see Section D), the 2D graph convolution can indeed serve as a good initialization.

F More Comments and Initialization Studies

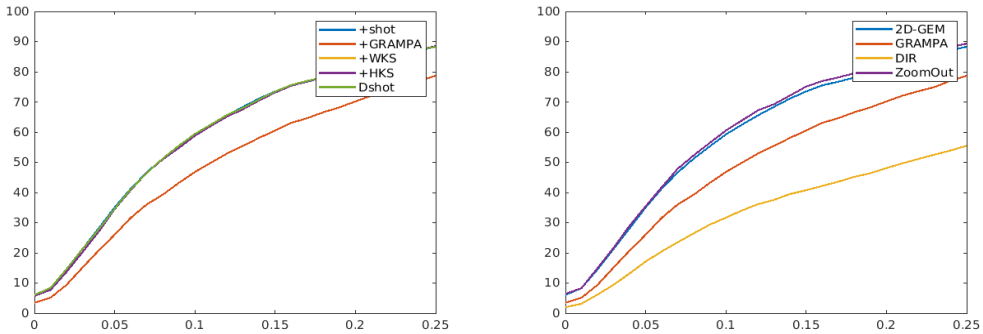
In this section we add some initialization studies for 2D-GEM 1. For these studies we used SCAPE, TOPKIDS and TOSCA-re-meshed [55], where TOSCA-re-meshed is the TOSCA dataset with each shape re-meshed to 5,000 vertices.



(a) Initializations on TOPKIDS (b) Initializations on SCAPE

Figure 9: 2D-GEM with different initializations on a sample shape from each dataset, i.e., TOPKIDS, SCAPE

As shown in Appendix D, 2D-GEM is robust to initializations on graphs and on shapes. When no features or initialization are present, 2D-GEM is shown to be comparable with the spectral baseline GRAMPA. However, when an informative initialization or features is present, 2D-GEM performs significantly better. On 3D-shapes, 2D-GEm tends to maintain a good performance despite the initializations as show in Figures 9(a), 9(b), and 10(a). This means that 2D-GEM is robust to initializations on 3D-shapes. Moreover, 2D-GEM itself following equation **??**. An interesting



(a) Initializations on TOSCA-re-meshed (b) Comparison on sample on TOSCA-re-meshed

Figure 10: 2D-GEM with different initializations on a sample shape from TOSCA-re-meshed as well as comparison with other methods on the same sample shape

topic for future work is an in-depth investigation of the re-meshed shapes as well as shapes with very different triangulation such as in SHRECK19 [50] (since a key component of 2D-GEM is the adjacency associated with the triangulation of the shapes).

For the plots in the figures, we used 100 eigen-vectors for the Heat Kernel Signature (HKS) and Wave Kernel Signatures (WKS). For GEM we used 500 eigen-vectors, maxiter 60, LMD 100 for all iterations on non-isometric sample from TOPKIDS, and LMD thresholds [0.5, 0.45, 0.4, 0.35, 0.3] for non-isometric shapes where we repeat 0.3 for the rest of the iterations beyond 5. DSHoT is directly using the shot feature (i.e., computing $X_A X_B^T$) and using 2D-GEM as in equation 4 to initialize the map and use this initialized map in the algorithm in Section 4. See Figures 10(a), 9(a) and 9(b).

G On Non-Smooth Final Maps Returned By 2D-GEM

Concerning the non-smooth final maps produced by the algorithm, we will indeed try to dive deeper into that in future extensions. However, for now, we note that it may arise due to the initial map being very poor and non smooth as pointed out by Consistent ZoomOut [26]. In most cases, this may be reduced to an extent by (a) either using 2D-GEM in a ZoomOut-like manner, i.e., starting with lower eigen-values and iteratively increasing them (though this strategy works in some cases, it may not always work as pointed out in [26]), or by (b) using more rounds of our pipeline, i.e., the LMD (since the LMD is an effective measure of the continuity and distortion of the map), the map update via convolution (for denoising the map), and the 2-Hop algorithm (since the 2-Hop algorithm also enforces neighbourhood preservation). See Figures 11 and 12 where the initial map given by shot was non-smooth.

H 2D-GEM's Sensitivity to LMD Thresholds

As noted in Section 5, the LMD thresholds differently for 2D-GEM on isometric shapes and on Isometric shape. Our choices of parameters were motivated by the properties of the shape. Namely, on non-isometric shapes where geometric properties are not preserved, we deactivate the constraint of enforcing preservation of geometric distortions by setting higher values for the LMD (i.e., updating the 2D spectrum with more anchor points) we used 100 in Section 5. On isometric shapes we set lower values, thereby enforcing this constraint. In this section, we conduct sensitivity studies on the LMD using an isometric pair of shapes, i.e., two random meshes from the SCAPE dataset as well as a non-isometric pair of shapes, i.e., two shapes from the TOPKIDS dataset.

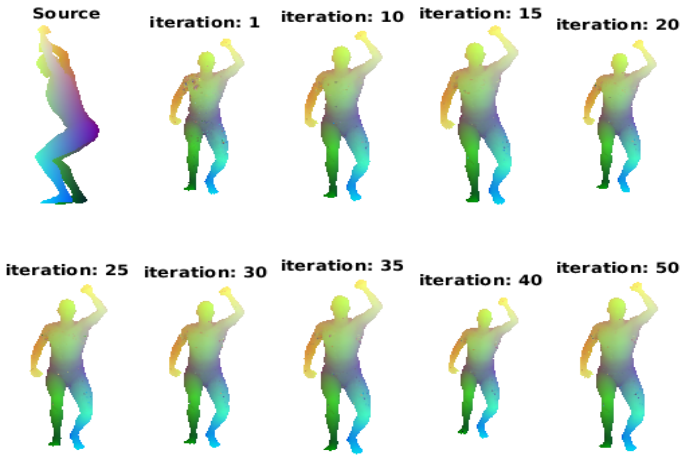


Figure 11: 2D-GEM's action on a non smooth initial map on SCAPE

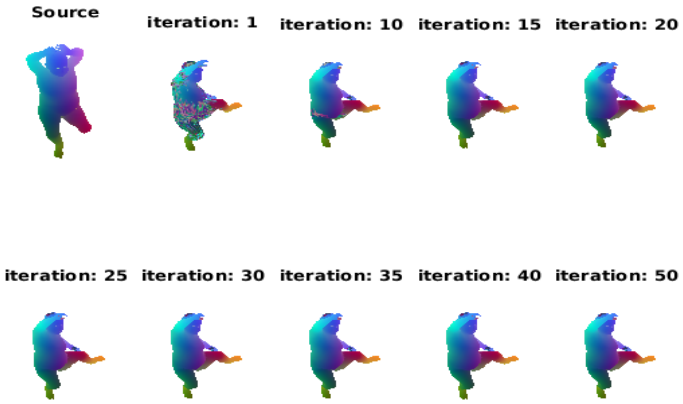
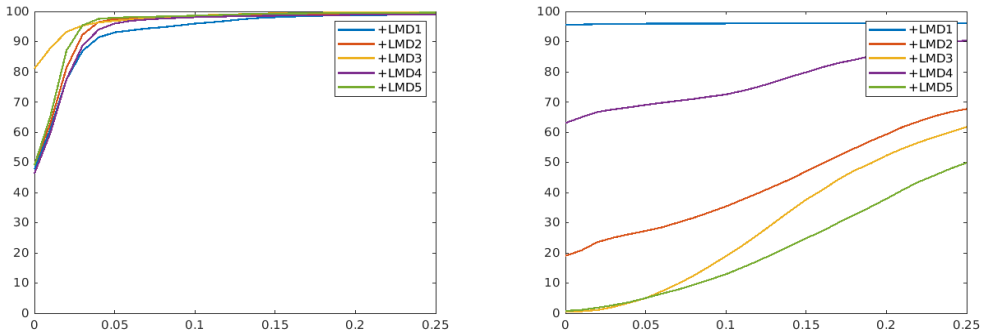


Figure 12: 2D-GEM's action on a non smooth initial map on TOPKIDS

For all figures in this section we used 60 iterations and set 5 different ranges of values for the LMD thresholds as follows:

- +LMD1 corresponds to using thresholds as $\epsilon_i = 100$ for all iterations.
- +LMD2 corresponds to using thresholds as ϵ_i =[100, 89, 78, 67, 56, 45, 34, 23, 12, 1] for the first 10 iterations and 1 for the rest of the iterations.
- +LMD3 corresponds to using thresholds as ϵ_i = [1, 1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1] for the first 10 iterations and 0.1 for the rest of the iterations.
- +LMD4 corresponds to using thresholds as ϵ_i = [5.00, 4.6667, 4.3333, 4.0000, 3.6667, 3.3333, 3.00, 2.6667, 2.3333, 2.00] for the first 10 iterations and 2.00 for the rest of the iterations.
- +LMD5 corresponds to using thresholds as ϵ_i = [3.00, 2.7333, 2.4667, 2.20, 1.9333, 1.6667, 1.40, 1.1333, 0.8667, 0.60] for the first 10 iterations and 0.60 for the rest of the iterations.

Figures 13(a) and 13(b) show that +LMD3 achieves 100% accuracy on isometric shapes. This shows that values from 1 and decreasing at a good rate after every iterations are good, which validates the



(a) LMD thresholds effects on 2D-GEM on SCAPE (b) LMD thresholds effects on 2D-GEM on TOPKIDS

Figure 13: 2D-GEM with different LMD on a sample shape SCAPE and a sample shape on TOPKIDS.

use of the constraint of preserving geometric properties for isometric shape matching (as shown in DIR[77]). On non isometric shapes, +LMD1 achieves the best accuracy, showing that geometric distortions are not preserved by non-isometrics, and so the LMD thresholds should be set high for these shapes.