ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

# Project Report for Complex Networks

*Heat Kernel Embedding and Graph Clustering*

Cristian Cioriia
Laurea Magistrale in Physics student
`cristian.cioriia@studio.unibo.it`

Department of Physics

August 25, 2022

## Abstract

This report explores geometrical algorithms that assess dissimilarity between 2d objects via the heat kernel embedding. Graph representations are generated based on equally separated 2d views of 3d objects, which are then subject to analysis with the goal of finding an embedding for which the separation between the different objects is most visible. Graph based curvature metrics such as sectional and gaussian curvature are computed using the graph laplacian representation. Hausdorff based distance measures are then used to gauge the similarity between graphs. Multidimensional scaling of the resulting distance matrices is performed, which allows for a simple 2d visualization of graph clustering.

# Contents

## 1 Introduction

Graph representations of images are attractive for purposes of segmentation, pattern recognition, and matching as they preserve key information of high dimensional data [1]. For instance, graphs preserve shape information which can be used to produce geometrical descriptions of the underlying images via curvature or extent (perimeter, area, volume etc.). Notwithstanding the fact that the transformation of an image into a graph is very particular to the convolutional filters (e.g. edge filters) used or feature extraction algorithms for most use cases a reasonably sized graph can be found. Furthermore, the low dimensionality of the graph allows for much faster processing in contrast with using the full information contained in an image.

Spectral graph theory has found much success in tasks such as object recognition and segmentation [2]. Much of what this program entails is first translating the graph structure into an adjacency matrix. However, given the fact that there is no canonical way of embedding a graph into a vector space [3], this raises issues when computing similarity measures. This is mainly because graphs have no standard ordering and more so graphs of differing number of nodes and edges can very well point to the same object of reference. One way to avoid this is to look at the dominating eigenvectors and use them to map graphs into a low dimensional space for clustering purposes. Roughly, this is the procedure of the algorithms, such as KPCA (Kernel Principal Component Analysis) or MDS (Multidimensional scaling) [2], that have had most success for computer vision tasks, in spite of the mentioned obstacles.

One of the most important methods of spectral graph theory is using the properties of the Laplcian matrix to distinguish between graphs. The Laplacian operator is found naturally by applying the heat diffusion equation to a graph. The kernel of the equation is the exponentiation of the Laplacian of the graph, which dictates how the flow of information propagates along the edges with respect to time. It is the spectrum of the Laplacian matrix which the work in this paper will concentrate on.

There has been work done to connect the spectral representation of a graph to geometry [1]. It has been shown that a graph can be viewed as residing on a manifold whose geodesic distances are characterized by the heat kernel. Motivated by this, this paper looks into how differential geometry concepts such as sectional curvature and gaussian curvature can be used as measures for graph matching [4]. This ultimately also assess the difference in efficiency between using Euclidean and geodesic distances. Once the two distinct sets of curvatures are computed, distance measures are used to gauge differences between different graphs. For this purpose, both classical Hausdorff [5] distance between sets of points and also modified Hausdorff distance will be used. Once the distance matrices are produced, the clustering is performed by first embedding the graphs into a low dimensional representation using Multidimensional scaling [6].

This paper is structured as follows. In section 2 an outline is made of the steps followed to perform the graph clustering. Section 3 is reserved for the analysis of the results and a discussion on the performance of the algorithm. Section 4 offers some conclusions on the work done in this report.

# 2 Theory

## 2.1 Spectral graph theory

Let $G = (V, E)$ be a graph, where $V$ is the set of nodes and $E \subseteq V \times V$ is the set of edges. The adjacency matrix $A$ is given by:

$$A(u,v) = \begin{cases} 1 & if(u,v) \in E \\ 0 & otherwise \end{cases} \tag{1}$$

In order to arrive at the Laplacian operator of a graph, first take $x_i$ as the set of probabilities of a system being in the $i-th$ state. Let $\mathrm{T_{ij}}$ be the transition matrix between states (from node j to node i ):

$$x_i(t+1) = \sum_j T_{ij} x_j(t) \quad or \tag{2}$$

$$\vec{x}(t+1) = T \cdot \vec{x}(t). \tag{3}$$

Given the adjacency matrix $A$, the matrix of transitions between states $T_{ij}$ is

$$T_{ij} = \frac{A_{ij}}{\sum_j A_{ij}} \quad or \quad T = A \cdot D^{-1}, \tag{4}$$

where D is the diagonal matrix of node degrees. Now it is easy to express the rate of change of the state with respect to time:

$$\frac{\vec{x}(t+1) - \vec{x}(t)}{1} = (I - T) \cdot \vec{x}(t) \quad or \quad \dot{\vec{x}} = L_T \cdot \vec{x}(t), \tag{5}$$

where $\mathrm{L_T} = (D - A) \cdot D^{-1}$ is the diffusion Laplace operator. The Laplacian matrix is just $L = D - A$, that is the degree matrix minus the adjacency matrix. Explicitly,

$$L(u,v) = \begin{cases} d_u & if u = v \\ -1 & if(u,v) \in E \\ 0 & otherwise. \end{cases} \tag{6}$$

The normalized Laplcian is given by $\widehat{L} = D^{-1/2} L D^{-1/2}$. The spectral decomposition of the normalized Laplacian matrix is

$$\widehat{L} = \Phi \Lambda \Phi^T = \sum_{i=1}^{|V|} \lambda_i \phi_i \phi_i^T, \tag{7}$$

where $|V|$ is the number of nodes and $\Lambda = diag\left(\lambda_1, \lambda_2, \ldots, \lambda_{|V|}\right)$ is the diagonal matrix with ordered eigenvalues as elements and $\Phi$ is the matrix with the eigenvectors as columns.

Ordering is assured because the Laplacian is a semi definite matrix with all eigenvalues being real and positive . The heat equation applied to a graph is thus expressed as

$$\left|\frac{\partial h_t}{\partial t}\right| = -\widehat{L}h_t, \tag{8}$$

where $h_t$ is the heat kernel and $t$ is the time. The solution is easily identified as a matrix exponential:

$$h_t = \exp[-\widehat{L}t] = \Phi\exp[-t\Lambda]\Phi^T. \tag{9}$$

## 2.2 Spectral graph geometry

The spectrum decomposition of the heat kernel allows for the fixing of the graph onto a vector space. The matrix containing the node coordinates $Y$ is obtained via Young-Householder decomposition $h_t = Y^T Y$. The $Y$ matrix can be written as:

$$Y = \left(y_1, y_2 \ldots, y_{|V|}\right) = \exp\left[-\frac{1}{2}t\Lambda\right]\Phi^T, \tag{10}$$

where $y_u$ is the coordinate vector for the node $u$. This representation allows for Euclidean distance to be computed between the nodes. For nodes $u$ and $v$

$$d_E^2(u,v) = (y_u - y_v)^T(y_u - y_v) = \sum_{i=1}^{|V|}\exp[-\lambda_i t](\phi_i(u) - \phi_i(v))^2. \tag{11}$$

If we are to look at geodesic distances, when a pair of nodes are connected by an edge, then $d_G(u,v) = 1$, otherwise it is equal to the number of edges of the shortest path connecting the 2 nodes.

### *2.2.1 Sectional Curvature*

We can leverage our intuition about geodesic and euclidean understanding to compute edge curvature. Given 2 nodes $u$ and $v$ between which there is an edge, we can think of them being fixed on manifold. We assume the geodesic between them to be an arc of a circle, and the euclidean distance between the nodes to be the chord of the arc. Take $r_s(u,v)$ to be the radius of the circle, and the tangent vector of the arc to change by $2\theta(u,v)$:

$$d_G(u,v) = 2r_s(u,v)\theta(u,v) \quad \textit{and} \quad d_E(u,v) = 2r_s(u,v)\sin\theta(u,v). \tag{12}$$

Using the Taylor expansion of the *sin* function, for small $\theta$, we can write:

$$d_E(u,v) = d_G(u,v) - \frac{d_G(u,v)^3}{24r_s^2(u,v)}. \tag{13}$$

Now we can express the radius in terms of the geodesic and euclidean distances:

$$k^2(u,v) = \frac{1}{r_s^2(u,v)} = d_G(u,v) - \frac{24\,(d_G(u,v) - d_E(u,v))}{d_G(u,v)^3}, \tag{14}$$

where $k(u,v)$ is the curvature associated to the edge. For an edge of the graph $d_G(u,v) = 1$ and thus

$$k^2(u,v) = 24\,(1 - d_E(u,v))\,. \tag{15}$$

Notice the euclidean distance associated to an edge can be at the maximum equal to the geodesic distance and thus it has an upper bound equal to unity. Equivalently, the curvature has a lower bound of 0.

*2.2.2 Gaussian curvature*

In contrast with the sectional curvature, the Gaussian curvature is computed over graph triangles. As a result, we assume the graph is a triangulation of section of a manifold $M$. Pick one triangle $\triangle_G$ on M, whose sides are geodesics, and define the interior angles to be $\alpha_i$ and the geodesic edge lengths $\{d_{Gi}\}_{i=1}^3$. Now consider the Euclidean triangle underpinned by this triangle. Take the edge lengths to be $\{d_{Ei}\}_{i=1}^3$ and interior angles to be $\{\varphi_i\}_{i=1}^3$. By averaging over all geodesic distances we take the triangle to live on a hypersphere with radius $\hat{r} = \frac{1}{3}\sum_{i=1}^3 r_i$, where $r_i$ correspond to the sectional radii of the individual edges. The area of the spherical triangle is given by:

$$F_G = \hat{r}^2 \int_{\triangle_G} K dM = \left(\sum_{i=1}^3 \alpha_i - \pi\right)\hat{r}^2, \tag{16}$$

where the integral is made over Riemannian volume element $dM$. We can thus express the radius of the sphere as

$$\frac{1}{\hat{r}^2} = \frac{\sum_{i=1}^3 \alpha_i - \pi}{F_G}. \tag{17}$$

The infinitesimal area of a spherical section is $dF = \hat{r}^2 \sin\theta d\theta d\varphi$, which integrated over $\theta$ and $\phi$ gives the area of the geodesic triangle as:

$$F_G = \frac{1}{2\hat{r}}\hat{d}_E^2. \tag{18}$$

where $\hat{d}_E^2$ is the average of the square of the Euclidean lengths of the edges corresponding to geodesics. Finally, the Gaussian curvature of the geodesic triangle is given by:

$$\hat{\kappa}_G \int_{\triangle_G} K dM = \frac{1}{2\hat{r}^3}\hat{d}_E^2 \quad . \tag{19}$$

## 2.3 Similarity measures

The Hausdorff distance is the most commonly used similarity metric. It allows gauging the distance between sets of unordered observations. The Hausdorff distance is defined over compact sets in a metric space. It is computed by taking all points of one set, computing the minimum over the distances to all points in the other set, and then taking a maximum over all these values. More formally, the classical Hausdorff distance between two finite point sets A and B is given by

$$H(A, B) = \max(h(A, B), h(B, A)), \tag{20}$$

where the directed Hausdorff distance from A to B is defined to be

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|, \tag{21}$$

where $\|.\|$ is some underlying norm on the points of A and B (e.g. Euclidean norm). Note this is always a positive value. However, the classical Hausdorff distance is very sensitive to outliers. Given 2 identical graphs apart from a node which is placed quite far away from the original position, the Hausdorff distance will be very large compared to other graphs of very different shapes. A more robust modified Hausdorff distance ($MHD$) is based on the average distance value instead of the maximum value:

$$h(A, B) = \frac{1}{N_A} \sum_{a \in A} \min_{b \in B} \|a - b\|. \tag{22}$$

These distances also have a graph representation. Consider two graphs $G_1 = (V_1, E_1, k_1)$ and $G_2 = (V_2, E_2, k_2)$, where $V_1, V_2$ are the sets of nodes, $E_1, E_2$ the sets of edges and $k_1, k_2$ the curvature matrices over either edges or triangles. The distances can then be defined as

1) The classic Hausdorff distance ($HD$) is

$$h_{HD}(G_1, G_2) = \max_{E_i \in E_1} \min_{E_j \in E_2} \|k_2(E_j) - k_1(E_i)\|. \tag{23}$$

2) The modified Hausdorff distance ($MHD$) is

$$h_{MHD}(G_1, G_2) = \frac{1}{|E_1|} \sum_{E_i \in E_1} \left( \min_{E_j \in E_2} \|k_2(E_j) - k_1(E_i)\| \right). \tag{24}$$

Note, there is an alternative definition over triangles where the underlying curvature is Gaussian. Furthermore, these distances are directional. In order for the graph distance matrix to be computed, the maximum between the distance in both directions $G_1 \to G_2$ and $G_2 \to G_1$ need to taken.

## 2.4 Multidimensional Scaling

At this point we have the distance matrix between graphs, which could act as a confusion matrix to gauge similarity. However, this is not good enough for clustering purposes.

What we need is a low dimensional representation of the distance matrix, which will reveal affinity in a way easily visualized. Classical Multidimensional Scaling (MDS) method is able to to embed the distance matrix in Euclidean space. Let $H$ be the distance matrix with row $r$ and coulmn $c$ entry $H_{rc}$. The MDS procedure then produces a matrix T whose element with row $r$ and column $c$ is given by:

$$T_{rc} = -\frac{1}{2}\left[H_{rc}^2 - \widehat{H}_{r.}^2 - \widehat{H}_{.c}^2 + \widehat{H}_{..}^2\right], \tag{25}$$

where

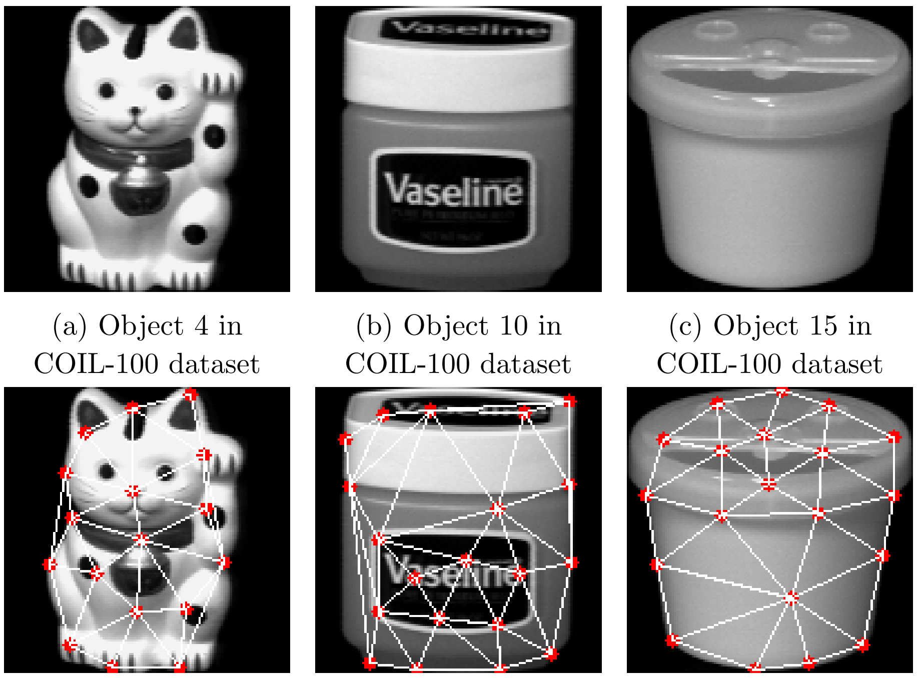
$$\widehat{H}_{r.} = \frac{1}{N}\sum_{c=1}^{N} H_{rc} \quad , \quad \widehat{H}_{c.} = \frac{1}{N}\sum_{r=1}^{N} H_{rc} \quad and \quad \widehat{H}_{..} = \frac{1}{N^2}\sum_{r=1}^{N}\sum_{c=1}^{N} H_{rc}. \tag{26}$$

The eigenvalue decomposition of $T$ is then used to obtain an embedding into Eucledian space. Given the $T$ matrix is not symmetric, the eigenvalues will be both positive and negative. In practice, we pick the dominating positive 2 or 3 eigenvalues and their corresponding eigenvectors. The embedding coordinate system for the graphs is $X = \left[\sqrt{l_1}u_1, \sqrt{l_2}u_2\right]$ where $l_1, l_2$ are the dominating eigenvalues and $u_1, u_2$ are the corresponding eigenvectors. The rows of this embedding are into one to one correspondence to the rows of the distance matrix. Hence, each row of the $X$ matrix corresponds to one graph, in the order they were present in the distance matrix.

## 3 Algorithm & Results

In order to fully arrive at the MDS visualization, the data was processed in the following way:

1. 3 objects of the *COIL-100* dataset were chosen randomly. 18 equally spaced views were selected for each object.
2. Shi-Thomas algorithm for feature extraction was used to find the best candidates for corners in each image. The algorithm was tuned such that about 20 nodes are created per image.
3. Delaunay triangulation was produced based on the nodes
4. The normalized Laplacian was computed from the adjancency matrix corresponding to the Delaunay triangulation of each image.
5. The Euclidean distance between the nodes of each graph was computed for different values of time e.g. $t \in (0.01, 0.1, 1, 10)$
6. Sectional and Gaussian curvatures corresponding to the edges and triangles of each graph was computed, as outlined in the Section 2.
7. A distance matrix between the graphs was computed using both the clssical Hausdorff (HD) distance and the modified Hausdorff distance (MHD).
8. Multidimensional scaling procedure was applied on the corresponding distance matrices to give the 2D representations displayed in the following figures.



(a) Object 4 in COIL-100 dataset (b) Object 10 in COIL-100 dataset (c) Object 15 in COIL-100 dataset

(d) Object 4 Delaunay triangulation (e) Object 10 Delaunay triangulation (f) Object 15 Delaunay triangulation

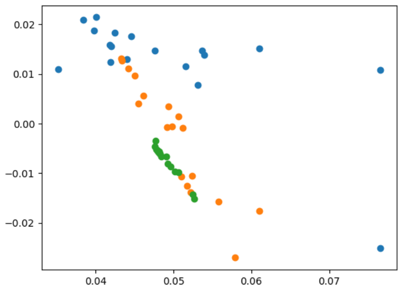
Figure 1: Objects used for the analysis in this report

| | | t = 0.01 | t = 0.1 | t= 1 | t = 10 |
|---|---|---|---|---|---|
| HD | Sectional Curvature | 0.203 | 0.388 | 0.611 | 0.611 |
| HD | Gaussian Curvature | 0.277 | 0.333 | 0.629 | 0.593 |
| MHD | Sectional Curvature | 0.260 | 0.296 | 0.277 | 0.259 |
| MHD | Gaussian Curvature | 0.370 | 0.333 | 0.185 | 0.537 |

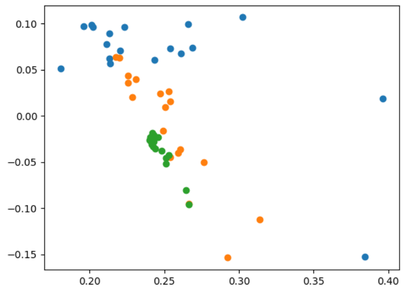
Table 1: Rand index table

In terms of algorithm complexity, the most time consuming task go into computing the Hausdorff distance matrices. If the average number of nodes is $n$, then computing the distance between two graphs has a time complexity of $O(n^2)$ and if there are $m$ graphs there should be $O(m^2)$ distances computed, as the distances are directional by Hausdorff measure. In this analysis there are 18 views per object and thus 54 images, and around 20 nodes per view. In Figure 2 there is noticeably no clear distinction between the different groups of poses for each object. The difference between the clusters does not improve significantly with varying time parameter. In order for the clustering efficiency to be more manifest, we use the Rand index. This is computed as follows:

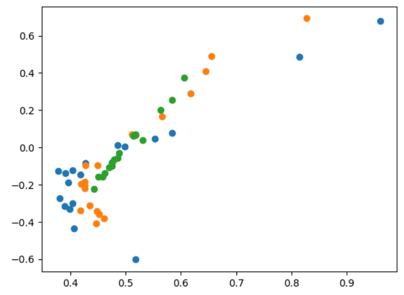
- Take the MDS representation and compute a mean for each group of poses
- Compute the distance from each graph to the mean of each group
- If the graph is closest to the mean of its own group then it is correctly placed in its cluster
- Rand index = # incorrect placements / # total graphs



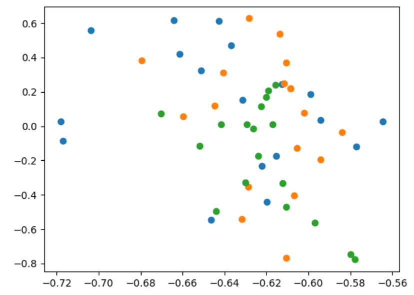
(a) t = 0.01



(b) t = 0.1

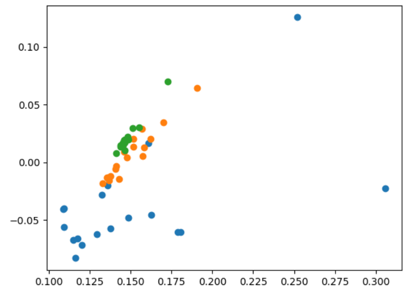


(c) t =1

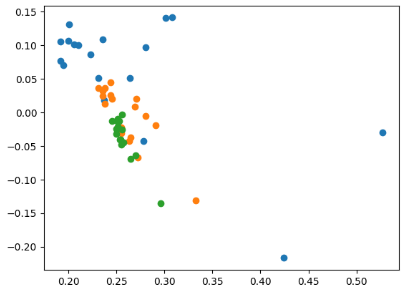


(d) t=10

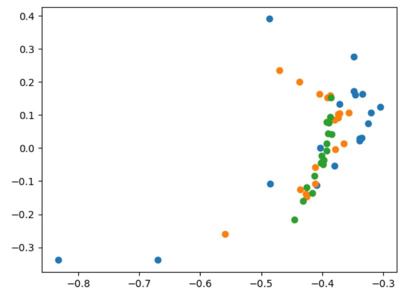
Figure 2: MDS representation based on the Classical Hausdorff distance using Gaussian curvatures as similarity measure. The colored groups correspond to poses of each of the 3 objects under analysis.



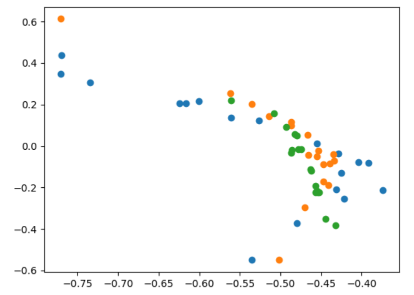
(a) t = 0.01



(b) t = 0.1



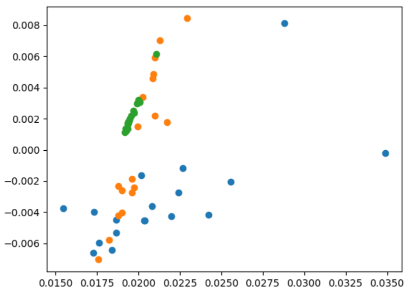
(c) t =1



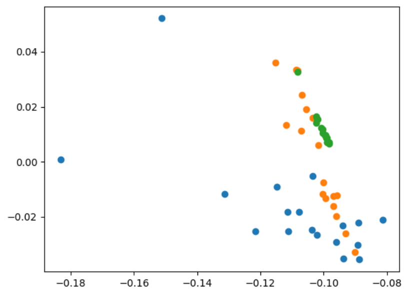
(d) t=10

Figure 3: MDS representation based on the classical Hausdorff distance using the Sectional curvature as similarity measure. The colored groups correspond to poses of each of the 3 objects under analysis.

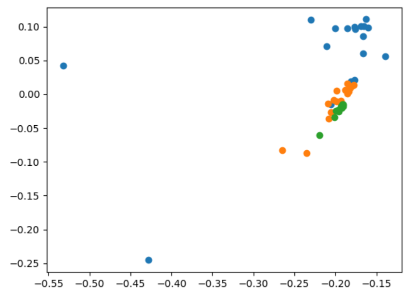
Based on the Rand index it is clear the modified Hausdorff distance improves the separation of the clusters, however it does not decrease the dissimilarity to a negligible amount. Note, the time parameter helps the clustering for values close to 1. Although the Gaussian curvature is a geometrical parameter better suited to gauging shape in 3d space, it does less well than the sectional curvature. This may be because the number of nodes for each graph is quite small, so the triangles cover quite a large area of the objects captured in the images. In addition to this, two of the objects have no less significant distinctive shape features and shadows play a less important role in defining the trinagulation. This may cause the curvatures matrices to look very similar irrespective of the rotation. Furthermore, the analysis has not included geodesic or euclidean distances between nodes that are more than 1 edge apart. In this way, a lot of the shape information of the graph has been ignored. This was taken as a sensible step, as the number of computations made over all combinations of nodes increases as the number of nodes to the second power and third power when triangles are taken into account. This means that for networks with $10^6$ nodes or more there should be $10^{18}$ triangles over which curvatures can be computed. Thus, only edges and triangles corresponding to the Delaunay triangulations have been considered. Possibly, a way to improve the Rand index would be to include also graph paths of length 2 or more. Out of all ways of gauging similarity the Sectional Curvature in tandem with the modified Hausdorff distance have performed the best.



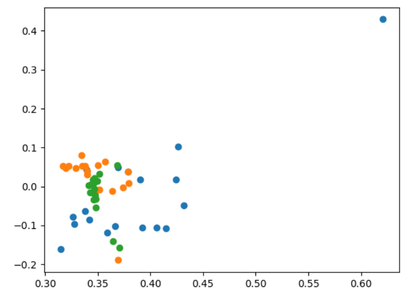
(a) t = 0.01



(b) t = 0.1

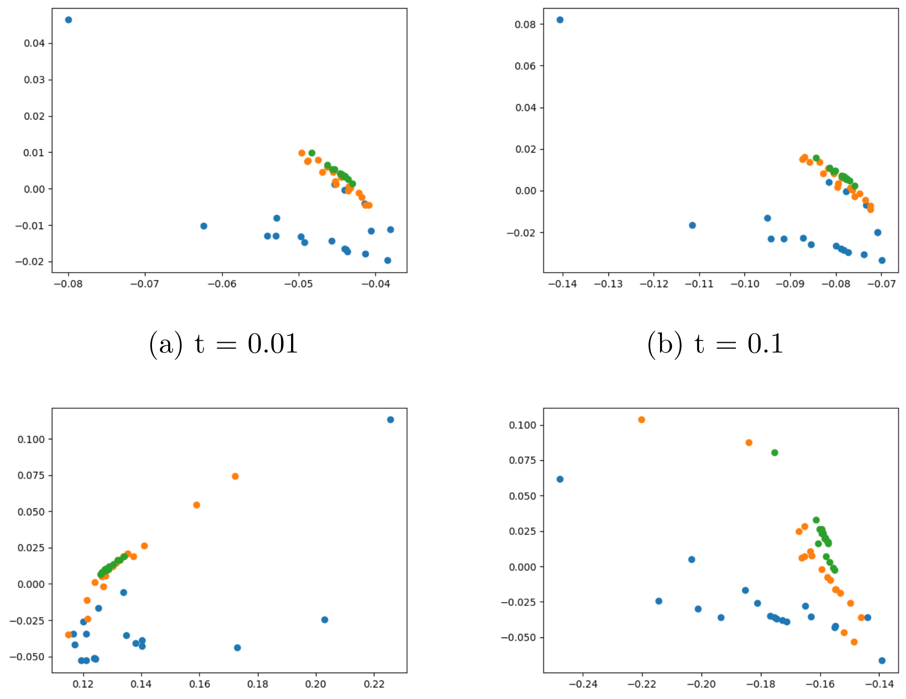


(c) t =1



(d) t=10

Figure 4: MDS representation based on the Modified Hausdorff distance using Gaussian curvatures as similarity measure. The colored groups correspond to poses of each of the 3 objects under analysis.



(a) t = 0.01

(b) t = 0.1

(c) t =1

(d) t=10

Figure 5: MDS representation based on the modified Hausdorff distance using Sectional curvatures as similarity measure.The colored groups correspond to poses of each of the 3 objects under analysis.

## 4 Conclusion

In this report a geometrical approach to graph clustering was investigated for the purposes of graph matching. The results agree with the findings of [2] when it comes to the more robust modified Hausdorff distance being a superior measure of gauging graph affinity. However, the Sectional curvature was found to perform better than the Gaussian curvature for both distance measures, in contrast to their findings. This can be due to the lack of consideration of longer paths on the graph, and triangles larger than the primary Delaunay triangulation. Also, the number of feature points of each graph may have been lower that the one in [2], however this is uncertain, as it is not specified in their report. Importantly, the embedding of the graph onto a manifold through the heat kernel representation has allowed us to leverage differential geometry concepts to gauge similarity between graphs. An improvement to the current work would be to use the current algorithm onto other datasets to check for robustness and to also add other measures such as perimeter, area and volume to have a more complete picture of similitude between graphs.

## References

[1] X. Bai and E. Hancock, "Heat kernels, manifolds and graph embedding," vol. 3138, 01 2004.

[2] H. ElGhawalby and E. R. Hancock, "Measuring graph similarity using spectral geometry," in *ICIAR*, 2008.

[3] B. Luo, R. C. Wilson, and E. R. Hancock, "Spectral embedding of graphs," *Pattern Recognition*, vol. 36, no. 10, pp. 2213–2230, 2003.

[4] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[5] M.-P. Dubuisson and A. Jain, "A modified hausdorff distance for object matching," in *Proceedings of 12th International Conference on Pattern Recognition*, vol. 1, pp. 566–568 vol.1, 1994.

[6] F. Wickelmaier, "An introduction to mds," 04 2003.