

# Data Clustering and Graph-Based Image Matching Methods

Yan Fang

Submitted for the degree of Doctor of Philosophy

The University of York  
Department of Computer Science

June 2012

## **Abstract**

This thesis describes our novel methods for data clustering, graph characterizing and image matching.

In Chapter 3, our main contribution is the M1NN agglomerative clustering method with a new parallel merging algorithm. A cluster characterizing quantity is derived from the path-based dissimilarity measure.

In Chapter 4, our main contribution is the modified log-likelihood model for quantitative clustering analysis. The energy of a graph is adopted to define the description length to measure the complexity of a clustering.

In Chapter 5, our main contribution is an image matching method based on Delaunay graph characterization and node selection. A normalized Euclidean distance on Delaunay graphs is found useful to estimate pairwise distances.

# Contents

|          |   |          |
|----------|---|----------|
| <b>1</b> | <b>Introduction</b>                                 | <b>1</b> |
| 1.1      | The Problems . . . . .                              | 1        |
| 1.2      | The Objectives . . . . .                            | 3        |
| 1.3      | Thesis Outline . . . . .                            | 4        |
| <b>2</b> | <b>Literature Review</b>                            | <b>5</b> |
| 2.1      | Data Clustering . . . . .                           | 5        |
| 2.1.1    | Hierarchical Clustering . . . . .                   | 5        |
| 2.1.2    | Partitional Clustering . . . . .                    | 8        |
| 2.1.3    | Clustering Methods . . . . .                        | 10       |
| 2.2      | Graph Representation . . . . .                      | 19       |
| 2.2.1    | Basic Graphs . . . . .                              | 20       |
| 2.2.2    | Matrix Representation . . . . .                     | 22       |
| 2.3      | Spectral Graph Theory . . . . .                     | 24       |
| 2.4      | Minimum Description Length . . . . .                | 25       |
| 2.5      | Graph Characterization and Node Selection . . . . . | 27       |
| 2.5.1    | Spectral Characterization of Graphs . . . . .       | 27       |
| 2.5.2    | Graph Characterization by Heat Kernel . . . . .     | 28       |
| 2.5.3    | Graph Characterization by Commute Time . . . . .    | 30       |
| 2.5.4    | Node Selection by Centralities . . . . .            | 31       |

|          |  |           |
|----------|--|-----------|
| 2.6      | Feature Detecting and Image Matching . . . . .           | 34        |
| 2.6.1    | Harris Corner Detecting . . . . .                        | 34        |
| 2.6.2    | SIFT Image Matching . . . . .                            | 36        |
| 2.7      | Summary . . . . .  | 37        |
| <b>3</b> | <b>M1NN Agglomerative Clustering</b>                     | <b>39</b> |
| 3.1      | M1NN Principle . . . . .                                 | 39        |
| 3.2      | Parallel Agglomerative Clustering . . . . .              | 42        |
| 3.2.1    | Classical Single-Link Agglomerative Clustering . . . . . | 42        |
| 3.2.2    | Parallel Clustering with M1NN Principle . . . . .        | 43        |
| 3.2.3    | MST-Based Parallel Clustering . . . . .                  | 44        |
| 3.3      | Path-Based Dissimilarity Measure . . . . .               | 45        |
| 3.4      | M1NN Agglomerative Clustering . . . . .                  | 50        |
| 3.4.1    | CP to CQ Expansion . . . . .                             | 50        |
| 3.4.2    | CQ Merging Rules . . . . .                               | 51        |
| 3.4.3    | M1NN Agglomerative Clustering with Strong Merging Rule   | 53        |
| 3.5      | Algorithm Description . . . . .                          | 55        |
| 3.6      | Experimental Analysis . . . . .                          | 57        |
| 3.6.1    | Clustering Process Demonstration . . . . .               | 57        |
| 3.6.2    | Chaining Phenomenon . . . . .                            | 57        |
| 3.6.3    | York2012 . . . . .                                       | 60        |
| 3.6.4    | Experimental Comparison . . . . .                        | 62        |
| 3.6.5    | Noise Test . . . . .                                     | 73        |
| 3.7      | Advanced Applications . . . . .                          | 73        |
| 3.7.1    | Data Mining . . . . .                                    | 77        |
| 3.7.2    | Image Segmentation . . . . .                             | 77        |
| 3.7.3    | Manifold Learning . . . . .                              | 81        |
| 3.8      | Conclusion . . . . .                                     | 82        |



|          |   |            |
|----------|---|------------|
| <b>4</b> | <b>Modified Log-likelihood Clustering</b>                             | <b>83</b>  |
| 4.1      | MDL Principle for Clustering . . . . .                                | 83         |
| 4.1.1    | Relationship to Log-likelihood Function . . . . .                     | 85         |
| 4.1.2    | Relationship to Graph Energy . . . . .                                | 85         |
| 4.2      | Algorithm Description . . . . .                                       | 89         |
| 4.3      | Experimental Analysis . . . . .                                       | 90         |
| 4.3.1    | Toy Graph Partitioning Selection . . . . .                            | 90         |
| 4.3.2    | Clustering Selection . . . . .  | 90         |
| 4.3.3    | Image Segmentation Comparison . . . . .                               | 91         |
| 4.3.4    | Clustering Method Comparison . . . . .                                | 91         |
| 4.3.5    | Further Discussion on MLL and ML . . . . .                            | 96         |
| 4.4      | Conclusion . . . . .  | 99         |
| <b>5</b> | <b>Delaunay Graph Characterization and Graph-Based Image Matching</b> | <b>100</b> |
| 5.1      | Heat Diffusion on Delaunay Graphs . . . . .                           | 100        |
| 5.1.1    | Critical Time of Heat Diffusion . . . . .                             | 101        |
| 5.1.2    | Delaunay Graph Characterization by Heat Diffusion . . . . .           | 102        |
| 5.2      | Graph-Based Image Matching . . . . .                                  | 107        |
| 5.2.1    | Delaunay Graph Selection for Image Matching . . . . .                 | 107        |
| 5.2.2    | Graph Node Selection for Image Matching . . . . .                     | 109        |
| 5.3      | Algorithm Description . . . . .                                       | 118        |
| 5.4      | Experimental Analysis . . . . .                                       | 118        |
| 5.5      | Conclusion . . . . .  | 124        |
| <b>6</b> | <b>Conclusions and Future Work</b>                                    | <b>125</b> |
| 6.1      | Contributions . . . . .   | 125        |
| 6.1.1    | M1NN Agglomerative Clustering . . . . .                               | 125        |
| 6.1.2    | Modified Log-likelihood Clustering . . . . .                          | 126        |

|       |   |     |
|-------|---|-----|
| 6.1.3 | Delaunay Graph Characterization and Graph-Based Image<br>Matching . . . . . | 127 |
| 6.2   | Limitations . . . . .   | 127 |
| 6.3   | Future Work . . . . .   | 129 |

# List of Figures

|      |  |    |
|------|--|----|
| 2.1  | Dendrogram example (from [66]) . . . . .                           | 11 |
| 2.2  | Responsibility and availability illustration (from [45]) . . . . . | 18 |
| 2.3  | A simple and undirected graph . . . . .                            | 20 |
| 2.4  | $k$ -NN graph example . . . . .                                    | 21 |
| 2.5  | Delaunay graph and Voronoi tessellation (from [88]) . . . . .      | 22 |
| 2.6  | Minimum spanning tree example (from [91]) . . . . .                | 22 |
| 2.7  | Graph and Laplacian matrix representation (from [93]) . . . . .    | 24 |
| 2.8  | MDL for linear regression problem . . . . .                        | 26 |
| 2.9  | SIFT descriptor representation (from [143]) . . . . .              | 37 |
| 2.10 | SIFT image matching example . . . . .                              | 37 |
| 3.1  | Observation 2 demonstration . . . . .                              | 40 |
| 3.2  | M1NN principle for clustering demonstration . . . . .              | 41 |
| 3.3  | CPs in a complex data set . . . . .                                | 42 |
| 3.4  | The problem of the classical agglomerative clustering . . . . .    | 43 |
| 3.5  | Dendrogram comparison of two clustering methods . . . . .          | 44 |
| 3.6  | Improper merging from parallel clustering . . . . .                | 45 |
| 3.7  | Segment removal demonstration . . . . .                            | 47 |
| 3.8  | Delaunay graph after removing triangles . . . . .                  | 48 |
| 3.9  | The bottleneck edges demonstration . . . . .                       | 49 |
| 3.10 | CP to CQ expansion . . . . .                                       | 51 |

|  |    |
|--|----|
| 3.11 Pseudocode of CQ merging algorithm . . . . .                | 54 |
| 3.12 CQ merging result . . . . .                                 | 55 |
| 3.13 Pseudocode of strong merging algorithm . . . . .            | 56 |
| 3.14 Strong merging result . . . . .                             | 56 |
| 3.15 Clustering demonstration 1 . . . . .                        | 58 |
| 3.16 Clustering demonstration 2 . . . . .                        | 59 |
| 3.17 Chaining phenomenon demonstration . . . . .                 | 60 |
| 3.18 York2012 clustering comparison 1 . . . . .                  | 63 |
| 3.19 York2012 clustering comparison 2 . . . . .                  | 64 |
| 3.20 York2012 clustering comparison 3 . . . . .                  | 65 |
| 3.21 Noisy York2012 clustering comparison . . . . .              | 66 |
| 3.22 Pairwise distance matrix (Shape25) for clustering . . . . . | 74 |
| 3.23 Clustering comparison for Shape25 . . . . .                 | 75 |
| 3.24 Clusterings of M1NN and DBSCAN under heavy noise . . . . .  | 75 |
| 3.25 Air travel clusters by AP . . . . .                         | 78 |
| 3.26 Key air travel clusters by M1NN . . . . .                   | 79 |
| 3.27 Image segmentation comparison . . . . .                     | 80 |
| 3.28 Manifold learning . . . . .                                 | 81 |
| 4.1 Graph energy demonstration . . . . .                         | 87 |
| 4.2 Graph energy inspection . . . . .                            | 88 |
| 4.3 Toy graph partitionings . . . . .                            | 91 |
| 4.4 KM clusterings on 7G (7 Gaussian clusters) . . . . .         | 92 |
| 4.5 KM clusterings on York2012 . . . . .                         | 93 |
| 4.6 Toy image segmentation . . . . .                             | 94 |
| 4.7 DL of KM clusterings on 7G . . . . .                         | 95 |
| 4.8 DL of KM clusterings on York2012 . . . . .                   | 95 |
| 4.9 DL of image segmentations . . . . .                          | 95 |

|      |  |     |
|------|--|-----|
| 4.10 | Bipartitioned toy graph 1 . . . . .                            | 97  |
| 4.11 | Bipartitioned toy graph 2 . . . . .                            | 97  |
| 5.1  | The distance matrices of NC1 . . . . .                         | 104 |
| 5.2  | The distance matrices of Complex8 . . . . .                    | 105 |
| 5.3  | Delaunay graphs with different $t_c$ on a face image . . . . . | 108 |
| 5.4  | Hot node evolution on a graph by increasing time . . . . .     | 111 |
| 5.5  | Hot node selection by using heat diffusion at $t_c$ . . . . .  | 112 |
| 5.6  | Node selection by using degree centrality . . . . .            | 113 |
| 5.7  | Node selection by using closeness centrality . . . . .         | 114 |
| 5.8  | Node selection by using betweenness centrality . . . . .       | 115 |
| 5.9  | Node selection by using eigenvector centrality . . . . .       | 116 |
| 5.10 | Node selection by using the dominant set . . . . .             | 117 |
| 5.11 | Image matching by our method . . . . .                         | 119 |
| 5.12 | Image matching by SIFT . . . . .                               | 120 |
| 5.13 | COIL-20 database (from [201]) . . . . .                        | 121 |
| 5.14 | COIL duck images (from [201]) . . . . .                        | 121 |
| 5.15 | Face matching comparison . . . . .                             | 122 |

# List of Tables

|     |   |     |
|-----|---|-----|
| 3.1 | The characteristics of data sets . . . . .          | 68  |
| 3.2 | RI of clustering methods . . . . .                  | 69  |
| 3.3 | MER of clustering methods . . . . .                 | 70  |
| 3.4 | RI of clustering methods . . . . .                  | 71  |
| 3.5 | MER of clustering methods . . . . .                 | 72  |
| 3.6 | Noise test between M1NN and DBSCAN . . . . .        | 76  |
| 4.1 | Description length for clustering methods . . . . . | 98  |
| 4.2 | Log-likelihood for clustering methods . . . . .     | 98  |
| 4.3 | MLL and ML comparison . . . . .                     | 98  |
| 5.1 | Critical times for the Delaunay graphs . . . . .    | 106 |
| 5.2 | Image matching average error comparison . . . . .   | 123 |

# Acknowledgments

I would like to thank my supervisor Professor Edwin Hancock for his support and suggestions on my PhD study.

I would like to thank my assessor Professor Richard Wilson for his patience and kindness in many long and enlightening conversations.

I would like to thank my colleagues and the department of computer science.

Finally, I really would like to thank my parents for their unconditional support and endless love.

# Declaration

I declare that all the work in this thesis is solely my own except where attributed and cited to another author.



# Chapter 1

## Introduction

In this chapter, the computer vision and pattern recognition problems studied in this thesis are briefly introduced. The motivation and objectives of our work are presented and followed by the thesis outline at the end of the chapter.

### 1.1 The Problems

Clustering is the process of organizing similar objects into groups. For example, apples and pears form different clusters. The *Golden Delicious* and *Red Delicious*, though not closely related [1], are both botanically more like an apple than a pear. In computer science, the aim of clustering is to discover such clusters from given data for further analysis. Clustering methods are widely used in image segmentation [2], information access and retrieval [3], genome study [4], etc.

There are two types of clustering problems: semi-supervised clustering and unsupervised clustering. Semi-supervised clustering contains a few auxiliary constraints, e.g. the pairwise *must-link* and *cannot-link* constraints between objects [5]. Unsupervised clustering is of an exploratory nature since data are unlabeled and unconstrained [6]. In this thesis, unsupervised clustering is researched. Un-

less stated otherwise, the term clustering means specifically unsupervised clustering.

As cluster analysis is fundamentally important for learning and understanding data, a large number of clustering methods had been proposed in the past. However some critical issues remain unresolved. The first and main problem is the lack of a single and general method that can cope with varied clustering tasks and unspecific data sets [7]. The second problem is that many current clustering algorithms are controlled by a set of internal parameters, which are decided empirically for optimal performance. When Jain reviewed the state-of-the-art data clustering methods in 2010, he commented, '*in spite of the fact that K-means was proposed over 50 years ago and thousands of clustering algorithms have been published since then, K-means is still widely used*' [8]. However, K-means can only provide very basic clustering for real applications.

Meanwhile, spectral graph theory had been used for clustering for a long time. As objects are represented by vertices of a graph and their relations are denoted by edges, a data clustering problem becomes a graph partitioning problem [9]. The best known spectral clustering method is probably the so-called normalized cut [2]. It is fast, accurate and is extended to image segmentation and other applications, but counterexamples alert us there is possibly an inside theoretical flaw as it cannot cope with some simple examples [10]. Besides using eigenvectors, the number of eigenvalues of magnitude 1 is equal to the number of clusters in an ideal case when clusters are infinitely separated. In a more complex and realistic situation, the eigenvalues are no longer reliable [11].

There are also problems with graph matching methods applying to image matching. Basically, image matching is the process of detecting and matching similar features on different images, e.g. edges, corners, ridges and valleys [12]. Unlike the methods using only coordinates or descriptors of feature points [13,

14], graph matching methods construct graphs on feature points and then match the graphs with advanced mathematical tools, e.g. spectral graph theory [15]. Ideally, robust image matching should be achieved by using this graph-based strategy. In practice however, an affine transformation could bring a radical change to the size or structure of a graph. An image matching method based on graph matching only cannot perform stably and accurately [16].

## 1.2 The Objectives

In this thesis, our main research interests are data clustering and image matching. For data clustering, we aim to design a single and general clustering method:

1. Compatible with any data types. For data points in two or three dimensions, our method works with the principle of gestalt perception [17] as the data points are collectively recognized having meaningful shapes by our vision system in this case [18].

2. Robust to outliers. Currently, very few clustering methods can deal with outliers effectively. On the other hand, it is very unlikely to get clean data without outliers. Our method will maximally retrieve true clusters whilst identify as many outliers as possible.

For graph-based image matching, we aim to:

1. Build Delaunay graphs on images by a new criterion. Critical time is introduced as a measure of the diameter of a graph to limit the number of feature points so that all graphs to be matched will have a similar size.

2. Characterize Delaunay graphs with heat diffusion. The heat kernel at the critical time captures stable graph partitions across a series of images. Matching these partitions is much easier and cheaper than matching whole graphs.

3. Assign image information to graph nodes for accurate matching. Many

graph-based methods use solely graph structures for matching and many image matching methods compare only feature descriptors. We add feature descriptors to selected graph nodes to enhance the performance.

### **1.3 Thesis Outline**

The remainder of the thesis is organized as follows: Chapter 2 reviews the literature. The background knowledge on clustering and graph characterizing are provided. Chapter 3 presents an adaptive hierarchical clustering method. Chapter 4 describes a clustering comparison method. Chapter 5 brings a graph-based image matching method. Chapter 6 summarizes the work in this thesis and the possible directions for future research.

# Chapter 2

## Literature Review

This chapter reviews the research literature on data clustering, graph characterization and image matching. The clustering and image processing algorithms widely used in science and industry are briefly introduced. The mathematical knowledge needed to understand this thesis is also included in this chapter.

### 2.1 Data Clustering

In the literature, data clustering methods can be divided into two types: hierarchical and partitional clustering [19]. For large scale data sets, these two types of methods are often integrated to improve processing speed and accuracy [20].

#### 2.1.1 Hierarchical Clustering

There are two strategies for hierarchical clustering. The first one is an agglomerative bottom-up strategy, which starts with taking each data point as a cluster and merges them successively until all the data points are in only one cluster. The second one is a divisive top-down strategy, which assigns all the data points to one cluster initially and splits each cluster into smaller ones recursively [21].

For any clustering method, a distance measure is needed to quantify the similarity of objects. For hierarchical clustering methods, some popular metrics are used, e.g. Euclidean distance, squared Euclidean distance, Manhattan distance [22] and Mahalanobis distance [23]. For text or non-numeric data, Hamming distance [24] and Levenshtein distance [25] are preferred [26].

A linkage criterion is also required to define the distance between clusters. A linkage is normally a function on the pairwise distances of the data points from different clusters. Linkage options include single-linkage [27], complete-linkage [28] or average-linkage [29]. Single-linkage is massively used for clustering as it is capable with non-isotropic clusters [27]. But it has a drawback called the *chaining phenomenon* [30]: two clusters may be joined together because of some scattered points between them, even there are many more points well separated from one cluster to the other. As a result, clustering may contain just a few large but incorrect clusters and many trivial ones.

Developed on the classical hierarchical clustering algorithm, BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [31] uses a hierarchical data structure called CF-tree to partition data points. BIRCH can find a good clustering with a single scan of the data. However as BIRCH uses a diameter parameter to control the boundary of a cluster, data including non-spherical clusters are not supported very well. It is also inconvenient that BIRCH may generate different clusters for the same data input in different orders.

CURE (Clustering Using REpresentatives) [32] is another hierarchical clustering method. CURE uses a constant number of points inside a cluster to represent that cluster. The similarity of two clusters is measured by the smallest distance between two representative points from each cluster. CURE can find clusters of arbitrary shape and size but cannot be applied directly to large data sets.

CHAMELEON [33] is a two-phase clustering method. CHAMELEON begins

with a  $k$ -NN graph [34] constructed on data points. The data points are then assigned to a large number of relatively small sub-clusters by graph partitioning algorithms. The small sub-clusters are merged using a dynamic framework. CHAMELEON is generally recognized as a better clustering method than BIRCH or CURE [35].

More recently, a novel hierarchical clustering method was developed on lossy data compression [36]. The cluster structure is characterized by coding length. At each iteration, two clusters are merged to maximize the decrease of the coding length. The process terminates when the coding length cannot be reduced by merging any pair of clusters. Although this method is similar to the classical agglomerative clustering method, its performance is significantly improved by using coding length as a new distance measure between clusters. However it is assumed that data are drawn from multiple low-dimensional linear or affine subspaces, other types of data are not supported [37].

To cluster complex data, Zell was proposed as a hierarchical method based on Zeta function of graphs [38]. A cluster descriptor is defined as the integration of all cycles in a cluster. The popularity of a cluster is conceptualized as the global fusion of variations of a cluster descriptor and is computed by means of the *leave-one-out strategy*. At each so-called Zeta merging iteration, two clusters with the maximum incremental popularity are merged. As Zell uses a directed  $k$ -NN graph, choosing the 'right'  $k$  is obviously very important. Otherwise the number of cycles may be too small to characterize a cluster. Zell also requires manually setting three other internal parameters, which limits its use for general cluster analysis.

## 2.1.2 Partitional Clustering

A partitional clustering method returns a set of flat partitions rather than a nested structure. Usually a partitional method generates clusters by minimizing an objective function. K-means as an old and simple partitional method [39], where  $k$  denotes the number of clusters, is still widely used for cluster analysis. However there are some problems with K-means. For example, an inappropriate  $k$  could result poor clustering. And there is not a guaranteed globally optimal solution for K-means. The clustering process is also dependent on initial means [40]. Furthermore, concentric clusters are not detectable because it is assumed clusters are spherical and well separated [41].

Fuzzy C-means soft clustering method [42] improves K-means by associating every data point with all the clusters using a membership function. The characteristic points are defined in every cluster, which are considered as the centers of the clusters. The core algorithm of C-means is actually an EM (Expectation-Maximization) algorithm [43]. An EM algorithm starts with guessing a number of parameters. The clustering probabilities of a data point are calculated from these parameters. Then the parameters are re-estimated from the probabilities. Since this process usually repeats for hundreds or thousands of times, C-means is computationally expensive to use and is affected by the *over-fitting problem* [44].

As both K-means and C-means are sensitive to the random initialization of means, AP (Affinity Propagation) [45] was proposed to address this issue. Concisely, AP views each data point as an exemplar at the beginning of clustering. Exemplar-to-member and member-to-exemplar messages transmit competitively among all data points until stable exemplars emerge. The exemplars and their members form the clusters. AP is capable of clustering large scale data at a high computational cost.

Lately, AP is extended to WAP (Weighted AP) [46] by integrating neighbor-



ing points together and keeping spatial structure between them. WAP generates identical clustering as AP but the empirical complexity of WAP is much lower [47]. Based on WAP, HI-AP as a hierarchical clustering method was proposed [48]. The basic idea of HI-AP is partitioning data into subsets randomly and then launching AP on each subset to find exemplars. The exemplars are processed by WAP to further construct a hierarchy.

In partitional clustering, there is another group of methods detecting clusters based on the density of data points in a region [49]. DBSCAN [50], as the best known density-based clustering method, is capable of discovering clusters of arbitrary shape. DBSCAN also classifies data points as core points, border points and noise points, which is quite useful for data analysis. To deal with the data containing both numerical and categorical attributes, DBSCAN was further generalized to GDBSCAN [51].

Both DBSCAN and GDBSCAN need to tune a set of internal parameters. To tackle this problem, nonparametric DBCLASD (Distribution Based Clustering of Large Spatial Databases) was proposed [52]. DBCLASD assumes data points are uniformly distributed in every cluster. An initial cluster is incrementally augmented by its neighboring points as long as the nearest neighbor distance set of the cluster still fits the expected distance distribution.

OPTICS (Ordering Points To Identify the Clustering Structure) [53] is another density-based clustering method specialized in clustering data of varying density. In this method, data points are ordered so that spatially closest points become neighbors to each other. Additionally, a special distance representing the density is stored for each data point. Two data points will belong to a same cluster only if their distance is accepted by the cluster. OPTICS is also used for building hierarchical clustering methods [54].

Graph-theoretic approaches are intensively studied for partitional clustering

as well. The earliest attempt [55] built an MST (Minimum Spanning Tree) to bipartition data points. In 1973, the *Fiedler vector* of a graph Laplacian was found useful for graph partitioning problems [56]. The discovery bloomed a family of graph-theoretic clustering methods based on eigenvectors and eigenvalues, which are called spectral clustering. The best known spectral clustering method is the normalized cut [2]. Later the normalized cut was developed with the Markov random walk [57]. Meanwhile a multi-class normalized cut was also proposed [58], in which  $k$  largest eigenvectors of a transition matrix are used to embed data points into an eigen-space and K-means is invoked to cluster the data points.

The major drawbacks of spectral clustering are running speed and numerical stability. The isoperimetric method [59] was proposed as a fast and stable substitute. Solving an optimal partitioning problem is transformed to minimizing the isoperimetric number of a graph. The main problem of this method is the selection of a ground node. There are some suggested grounding strategies, e.g. grounding the maximum degree node or a random node [60].

More recently, a new graph-theoretic approach [61] for pairwise clustering was developed on the analogy between a cluster and a *dominant set* of vertices. The dominant set of a graph generalizes the maximal complete subgraph of an unweighted graph. It can be computed straightforward using continuous optimization techniques called replicator dynamics [62]. This method can also be used for hierarchical clustering [63].

### 2.1.3 Clustering Methods

In this section, some classical and effective clustering methods are introduced. For each method, a brief step-by-step algorithm is also given as a reference.

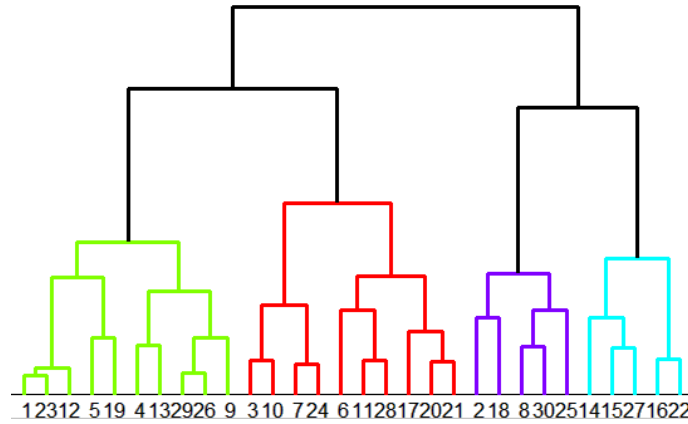


FIGURE 2.1: DENDROGRAM EXAMPLE (FROM [66])

### Single-Link Agglomerative Clustering

The single-link agglomerative clustering method may be the simplest and most intuitive hierarchical clustering method. In this method, the linkage does not have to be distance based. A variance increase (Ward's method [64]) or marginal likelihood [65] can also be adopted. When the Euclidean distance is used as the metric, the single-linkage for two sets  $A$  and  $B$  is defined as

$$sl(A, B) = \min\{d(a, b) | a \in A, b \in B\} \quad (2.1)$$

where  $d(a, b)$  denotes the Euclidean distance between points  $a$  and  $b$ .

Basically clusters are formed in a greedy manner. In Figure 2.1, the clustering process is demonstrated by a *dendrogram* [66].

The single-link clustering algorithm is given as:

1. Set each data point as its own cluster and compute a single-linkage matrix.
2. Merge two clusters with the minimum single-linkage and update the single-linkage matrix.
3. Repeat step 2 until all data points are grouped into one cluster.

## K-means and K-medoids

Given  $n$  data points  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  in the same dimensions and  $k$  clusters  $\Omega = (w_1, w_2, \dots, w_k)$ , K-means aims to minimize the within-cluster sum of squares

$$\arg \min_{\Omega} \sum_{i=1}^k \sum_{x_j \in w_i} \|x_j - \mu_i\|^2 \quad (2.2)$$

where  $\mu_i$  is the mean of points in  $w_i$ .

The algorithm of K-means combines an assignment step and an update step. It is an iterative refinement approach and will converge to a local minimum as minimizing the objective function is NP-hard even for  $k = 2$  [67]. A concise description of the algorithm [68] is given as:

1. Initialization. Set  $k$  means to random values  $m_1^{(1)}, m_2^{(1)}, \dots, m_k^{(1)}$ .
2. Assignment step. Each data point is assigned to the nearest mean

$$w_i^{(t)} = \{x_p : \|x_p - m_i^{(t)}\| \leq \|x_p - m_j^{(t)}\| \forall 1 \leq j \leq k\} \quad (2.3)$$

3. Update step. Calculate the new mean as the centroid of the data points in a cluster

$$m_i^{(t+1)} = \frac{1}{|w_i^{(t)}|} \sum_{x_j \in w_i^{(t)}} x_j \quad (2.4)$$

4. Repeat step 2 and 3 until assignments do not change.

Because of the randomness in initialization, K-means may not give the same clustering each time. In practice, K-means is normally run for multiple times for a possibly right clustering [69].

K-medoids is another partitioning clustering method quite similar to K-means. The main difference is the centers (medoids or exemplars) have to be on the data points [70]. K-medoids is more robust to noise and outliers than K-means but is computationally more expensive.

## DBSCAN

DBSCAN (Density Based Spatial Clustering of Applications with Noise) is a density based data clustering method proposed in 1996 [50]. The inspiration is from the observation that inside each spatial cluster, the density of points is considerably higher than outside. A cluster is defined by DBSCAN on the notion of density reachability. There are two parameters required: an  $\epsilon$  or  $k$  to define the neighborhood size and a *MinPts* to control the minimum number of points in a cluster. Given distance  $d(a, b)$  between points  $a$  and  $b$ ,  $a$  is said to be directly density reachable from  $b$  if satisfying  $d(a, b) < \epsilon$ . Otherwise if they are not directly reachable, but there is a sequence of points  $\{a, p_1, p_2, \dots, p_n, b\}$  that  $p_{i+1}$  and  $p_i$  are directly reachable, then  $a$  and  $b$  are defined as being density reachable to each other. In a cluster, all the data points are considered mutually density reachable and connected.

The algorithm is given as:

1. Start with an arbitrary unvisited point  $p$ .
2. Compare the parameter *MinPts* with the number of data points in  $p$ 's  $\epsilon$ -neighborhood, if *MinPts* is smaller,  $p$  and its neighborhood are identified as a cluster; otherwise,  $p$  is regarded as a noise point.
3. Run step 2 on all the neighborhood points of  $p$  and their neighborhood points and so on until no new cluster is generated. All the clusters originated from  $p$  are merged as one.
4. Repeat step 1 until all the data points are visited.

## Normalized Cut

The normalized cut method is probably the best known spectral clustering method based on a new graph-theoretic criterion [2]. The minimization of the normalized cut is formulated as a generalized eigenvalue problem.

Given a graph  $G(V, E)$ , it can be partitioned into two disjoint sets  $A$  and  $B$ , i.e.  $A \cup B = V$  and  $A \cap B = \emptyset$ . In this case, the edges connecting  $A$  and  $B$  are removed and the total weight of the edges is called a *cut*

$$cut(A, B) = \sum_{a \in A, b \in B} w_{(a,b)} \quad (2.5)$$

Finding the minimum cut corresponds to optimally bipartitioning a graph. However the minimum cut favors cutting small sets of isolated vertices [71]. To solve this problem, the normalized cut is defined as

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} \quad (2.6)$$

where  $assoc(A, V) = \sum_{a \in A, v \in V} w_{(a,v)}$  is the total association from nodes in set  $A$  to all the nodes in the whole graph.

Directly minimizing the normalized cut is NP-complete [72]. So the definition (2.6) is rearranged and the objective is changed to minimize the scalar expression

$$\arg \min_y \frac{\mathbf{y}^T (\mathbf{D} - \mathbf{A}) \mathbf{y}}{\mathbf{y}^T \mathbf{D} \mathbf{y}} \quad (2.7)$$

subject to  $\mathbf{y}(i) \in \{1, -b\}$  and  $\mathbf{y}^T \mathbf{D} \mathbf{1} = 0$ .  $\mathbf{y}(i)$  denotes the  $i$ -th element of vector  $\mathbf{y}$ ,  $b = \frac{\sum_{i \in A} deg(i)}{\sum_{j \in B} deg(j)}$  and  $\mathbf{1}$  is the  $|V| \times 1$  vector of all ones.  $\mathbf{D}$  is the degree matrix and  $\mathbf{A}$  is the adjacency matrix.

The expression (2.7) is actually the *Rayleigh quotient* [73]. If  $\mathbf{y}$  is relaxed to real values, it is equivalent to solving a generalized eigenvalue problem

$$(\mathbf{D} - \mathbf{A}) \mathbf{y} = \lambda \mathbf{D} \mathbf{y} \quad (2.8)$$

by  $\mathbf{L} = \mathbf{D} - \mathbf{A}$ , it can be rewritten as

$$\mathbf{L} \mathbf{y} = \lambda \mathbf{D} \mathbf{y} \quad (2.9)$$

defining  $\mathbf{y} = \mathbf{D}^{-\frac{1}{2}} \mathbf{z}$ , then we have

$$\mathbf{D}^{-\frac{1}{2}} \mathbf{L} \mathbf{D}^{-\frac{1}{2}} \mathbf{z} = \lambda \mathbf{z} \quad (2.10)$$

where  $\mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$  is called the *normalized Laplacian* and is denoted by  $\mathcal{L}$ , so we have

$$\mathcal{L}\mathbf{z} = \lambda\mathbf{z} \quad (2.11)$$

Comparing with Equation (2.31),  $\mathbf{z}$  is indeed the eigenvectors of the normalized Laplacian  $\mathcal{L}$ . The smallest eigenvector  $\mathbf{z}_0 = \mathbf{D}^{\frac{1}{2}}\mathbf{1}$  corresponds to  $\lambda_0 = 0$ . The constraint  $\mathbf{y}^T\mathbf{D}\mathbf{1} = 0$  can be rewritten as  $\mathbf{z}^T\mathbf{z}_0 = 0$ , from which we can deduce the range of  $\mathbf{y}$  as  $\{\mathbf{D}^{-\frac{1}{2}}\mathbf{z}_1, \mathbf{D}^{-\frac{1}{2}}\mathbf{z}_2, \dots, \mathbf{D}^{-\frac{1}{2}}\mathbf{z}_n\}$ .

Replacing  $\mathbf{y}$  with  $\mathbf{z}$  in (2.7), it becomes minimizing the scalar expression

$$\arg \min_z \frac{\mathbf{z}^T \mathcal{L} \mathbf{z}}{\mathbf{z}^T \mathbf{z}} \quad (2.12)$$

The solution is the second smallest eigenvector  $\mathbf{z}_1$  corresponding to the smallest non-zero value  $\lambda_1$ , which is similar to the *Fiedler vector* of an unnormalized Laplacian [74]. The original constraint  $\mathbf{y}(i) \in \{1, -b\}$  is relaxed to  $\mathbf{y}_1 = \mathbf{D}^{-\frac{1}{2}}\mathbf{z}_1$  to include real valued elements.

The normalized cut clustering algorithm is given as [2]:

1. Build a graph and compute the pairwise similarity matrix  $\mathbf{A}$  for given data.
2. Solve  $(\mathbf{D} - \mathbf{A})\mathbf{x} = \lambda\mathbf{D}\mathbf{x}$  for eigenvectors.
3. Use the second smallest eigenvector to bipartition the graph.
4. Recursively subdivide the segments if necessary.

### Isoperimetric Partitioning

The isoperimetric partitioning method was proposed to compete with other global partitioning methods in terms of quality [59]. It is motivated by the *isoperimetric constant* on continuous manifolds [75]. The isoperimetric constant  $h$  of a manifold is defined by Cheeger as

$$h = \inf_S \frac{|\partial S|}{\text{Vol}_S} \quad (2.13)$$

where  $S$  is a region in the manifold,  $\text{Vol}$  is the volume of the region and  $|\partial S|$  is the area of the boundary of the region. Hence  $h$  is the infimum of the ratio over all possible regions in the manifold.

In finite graphs, this constant is termed the *isoperimetric number*  $h_G$  [76] and is defined as

$$h_G = \min_S \frac{|\partial S|}{\text{Vol}_S} \quad (2.14)$$

where  $|\partial S|$  is actually the  $\text{cut}(S, \bar{S}) = \sum_{i \in S, j \in \bar{S}} w_{(i,j)}$ .  $\text{Vol}_S = |S|$  is the combinatorial volume.

Given a set of nodes on a graph, the ratio of its boundary to its volume is called the *isoperimetric ratio*. The isoperimetric partitioning method aims to find the partitions with the minimum isoperimetric ratio.

By defining an indicator variable  $x$ , which is 0 when node  $i$  is in  $S$  and 1 when  $i$  is in  $\bar{S}$ , we have

$$|\partial S| = x^T L x \quad (2.15)$$

where  $L$  is the Laplacian matrix. As we also have  $\text{Vol}_S = x^T \mathbf{1}$ , where  $\mathbf{1}$  denotes a vector with all entries one, the isoperimetric ratio in (2.14) can be rewritten as

$$h_G = \min_x \frac{x^T L x}{x^T \mathbf{1}} \quad (2.16)$$

By setting  $\text{Vol}_S = x^T \mathbf{1} = k$ , relaxing  $x$  to real nonnegative values and introducing a Lagrange multiplier  $\mu$  [77], it is equivalent to minimizing the function

$$Q(x) = x^T L x - \mu(x^T \mathbf{1} - k) \quad (2.17)$$

Differentiating  $Q(x)$  against  $x$  and setting the derivative to zero yield

$$2Lx = \mu \mathbf{1} \quad (2.18)$$

Ignoring all scalars does not affect  $x$  as a partitioning indicator, so we have  $Lx = \mathbf{1}$ . Because  $L$  is singular, its inverse is undefined. If we arbitrarily designate



a node  $v_g$  to be included in  $S$ , then  $x(g) = 0$ . Thus we can remove the  $g$ -th row and column of  $L$  and have

$$L_0 x_0 = \mathbf{1}_0 \quad (2.19)$$

where  $x_0$  and  $\mathbf{1}_0$  denote  $x$  and  $\mathbf{1}$  with their  $g$ -th elements being removed.

Equation (2.19) is now a nonsingular system. The partitioning indicator can be obtained by solving  $x_0$  plus  $x(g) = 0$ .

The isoperimetric partitioning algorithm is given as:

1. Choose a ground node  $v_g$ .
2. Solve  $L_0 x_0 = \mathbf{1}_0$  to have the indicator vector  $x = \{x_0, x(g)\}$ .
3. Cut  $x$  by a threshold to bipartition the graph.
4. Recursively subdivide the segments if necessary.

### Affinity Propagation

The affinity propagation method was published in *Science* 2007 [45] as a novel data clustering method. It takes all the data points as exemplars simultaneously at the beginning. Deterministic messages (affinities) are exchanged among the points until a number of exemplars gradually emerge to form clusters with the points around them. Unlike K-means, the affinity propagation method is a deterministic method [78].

The similarity between data points  $i$  and  $j$  is defined as a negative distance, i.e.  $s(i, j) = -d(i, j)$ . For each point  $k$ ,  $s(k, k)$  is referred as a priori preference that  $k$  is an exemplar. If we assume all points are equally suitable to be exemplars, we can initialize  $s(k, k)$  by the median or minimum value of all the pairwise similarities.

There are two kinds of messages exchanged between data points. The *responsibility*  $r(i, k)$  is sent from point  $i$  to the candidate exemplar point  $k$  to indicate how strongly  $i$  favors  $k$  over other candidate exemplars. The *availability*  $a(i, k)$  is

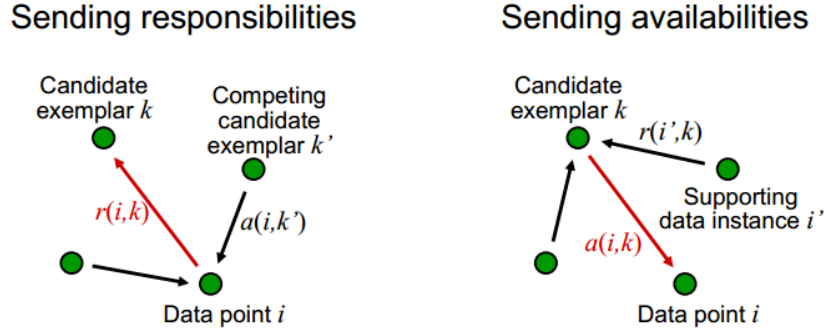


FIGURE 2.2: RESPONSIBILITY AND AVAILABILITY ILLUSTRATION (FROM [45])

sent from  $k$  to  $i$  to reflect how appropriately  $k$  will be the exemplar for  $i$ , taking account of the support from other points that  $k$  should be an exemplar. The two messages are demonstrated in Figure 2.2.

Initially we set  $a(i, k) = 0$ . The update rule of  $r(i, k)$  is

$$r(i, k) \leftarrow s(i, k) - \max_{k' \neq k} \{a(i, k') + s(i, k')\} \quad (2.20)$$

where in the first iteration,  $r(i, k)$  is in fact the similarity of  $i$  and  $k$  minus the largest similarity between  $i$  and other exemplars.

Then  $a(i, k)$  is updated by

$$a(i, k) \leftarrow \min \left\{ 0, r(k, k) + \sum_{i', i' \notin \{i, k\}} \max \{0, r(i', k)\} \right\} \quad (2.21)$$

The *self-availability*  $a(k, k)$  is updated in a different way

$$a(k, k) \leftarrow \sum_{i', i' \notin \{i, k\}} \max \{0, r(i', k)\} \quad (2.22)$$

During affinities propagating, availabilities and responsibilities are combined to identify exemplars. For example, if  $k$  maximizes  $a(i, k) + r(i, k)$ , then  $k$  is the exemplar for  $i$ .

In practice, unstable dynamics from numerical oscillation are avoided by damping messages

$$r(i, k)^* = \lambda r(i, k) + (1 - \lambda) r(i, k)^{\text{old}} \quad (2.23)$$

$$a(i, k)^* = \lambda a(i, k) + (1 - \lambda) a(i, k)^{\text{old}} \quad (2.24)$$

where  $\lambda = 0.5$  in the algorithm, and  $\lambda = 0.9$  in the implementation code [79].

The affinity propagation clustering algorithm is given as:

1. Build the similarity matrix using  $s(i, j) = -d(i, j)$  and  $s(k, k) = \min_{i,j} \{s(i, j)\}$ .
2. Initialize  $r(i, k) = 0$  and  $a(i, k) = 0$ .
3. Update all responsibilities using Equation (2.20).
4. Update all availabilities using Equation (2.21) and (2.22).
5. Combine the outcomes from step 3 and 4 to monitor exemplar decisions.
6. Repeat step 3, 4 and 5 for at least 10 times until the exemplar decisions are not changed.

## 2.2 Graph Representation

A graph is probably the most ubiquitous representation to reflect both natural and human-made structures. In computer science, a graph can conveniently represent networks of communication, data organization, computational devices, flows of computation and so on. It is defined on a collection of vertices and edges that connect pairs of vertices [80, 81], and is expressed mathematically as an ordered pair  $G = (V, E)$ .  $V$  denotes vertices and  $|V|$  is the number of vertices.  $E \subseteq V \times V$  stands for edges and  $|E|$  is the number of edges. For example, when we are given a graph in Figure 2.3, we will have  $V = \{1, 2, 3, 4, 5, 6\}$  and  $E = \{\{1, 2\}, \{1, 5\}, \{2, 3\}, \{2, 5\}, \{3, 4\}, \{4, 5\}, \{4, 6\}\}$ .

Precisely speaking, graphs can be distinguished as *finite* or *infinite*, *directed* or *undirected*, *simple* or *multiple*, *weighted* or *unweighted*, etc. Comparing with an infinite graph, a finite graph has countable vertices and edges. Edges in an undirected graph have no orientations, i.e.  $\{a, b\}$  is identical to  $\{b, a\}$ . On the other hand, in a directed graph  $\{a, b\}$  is not equal to  $\{b, a\}$ , both  $\{a, b\}$  and  $\{b, a\}$  may

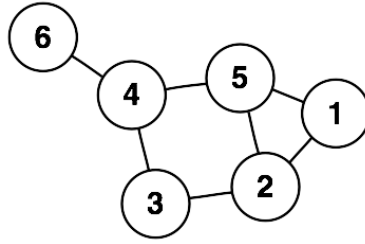


FIGURE 2.3: A SIMPLE AND UNDIRECTED GRAPH

coexist between  $a$  and  $b$ . A graph is simple if the graph is undirected without *loops* and there is only one edge between any two different vertices. A loop is an edge starting and ending on a same vertex [82]. A graph is weighted if a value other than 0 or 1 is assigned to an edge. The weight can present distance, time, cost, etc.

### 2.2.1 Basic Graphs

There are some very basic and commonly used graphs for computer vision and pattern recognition problems. For data points in  $n$ -dimensional vector space, i.e.  $\mathbf{a} = (a_1, a_2, \dots, a_n)$  or denoted by  $\mathbf{a} \in \mathbb{R}^n$ , a  $k$ -NN ( $k$ -Nearest Neighbor) graph is often used. A  $k$ -NN graph is built from connecting vertex  $u$  with  $v$  if  $v$  is one of  $k$  nearest neighbors of  $u$ . Because neighborhood relations are not always symmetric, some nodes may be linked with more than  $k$  nodes. A graph with only the edges connecting mutual neighbors is called a *mutual  $k$ -NN graph*, which is a subgraph of the corresponding  $k$ -NN graph.

A  $k$ -NN graph with  $k = 10$  is shown in Figure 2.4. The collection of red edges denotes the mutual 10-NN graph. The red and blue edges together correspond to the 10-NN graph. Generally a  $k$ -NN graph is not a planar graph [83]. If  $k = n - 1$ , where  $n$  is the total number of data points, we will have a **complete graph** or say fully connected graph. An  $\epsilon$ -neighborhood graph [84], defined similar to a  $k$ -NN

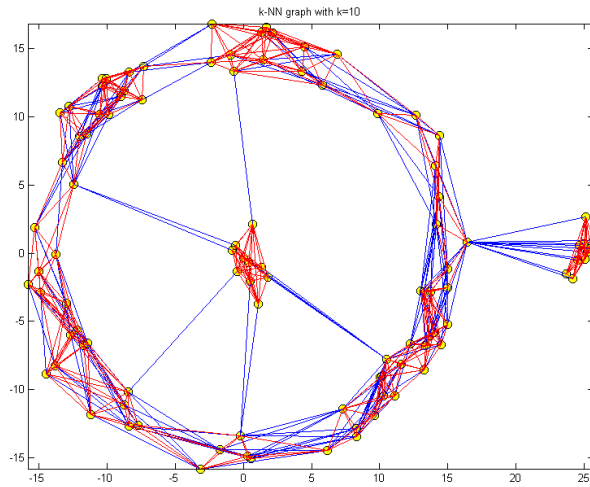


FIGURE 2.4:  $k$ -NN GRAPH EXAMPLE

graph, can also be seen in the literature.

A **Delaunay graph** [85] is another widely used graph. It is constructed by a Delaunay triangulation. In a Delaunay graph, there is no point inside the circum-circle of any triangles. To have a unique Delaunay graph, it is required all the nodes are in *general positions* [86], i.e. no three nodes lying on a same straight line and no four nodes on a same circle. Its dual graph is called a Voronoi tessellation [87] produced by connecting the centers of the circumcircles of triangles. In Figure 2.5, a Delaunay graph is plotted in black with all the circumcircles and their centers in red. A Voronoi tessellation is shown in red polygons overlapped the Delaunay graph.

Besides  $k$ -NN and Delaunay graphs, an **MST (Minimum Spanning Tree)** is also seen in many applications. In graph theory, a tree is defined as an undirected and connected graph without cycles [89]. There is only one path connecting each two vertices in a tree. A spanning tree is a subgraph connecting all the vertices of the original graph. One graph can have many spanning trees but only one MST

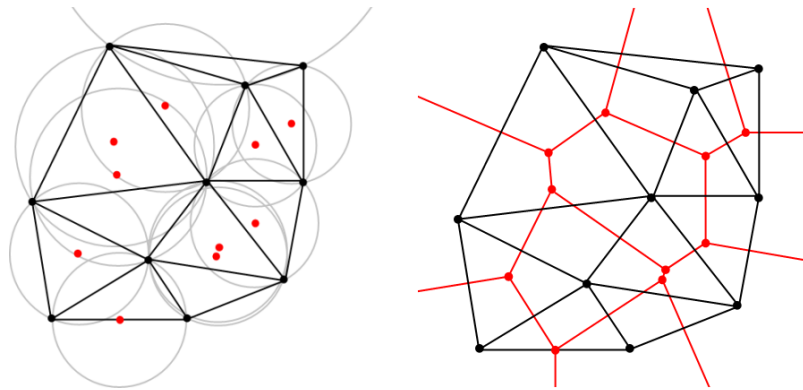


FIGURE 2.5: DELAUNAY GRAPH AND VORONOI TESSELLATION (FROM [88])

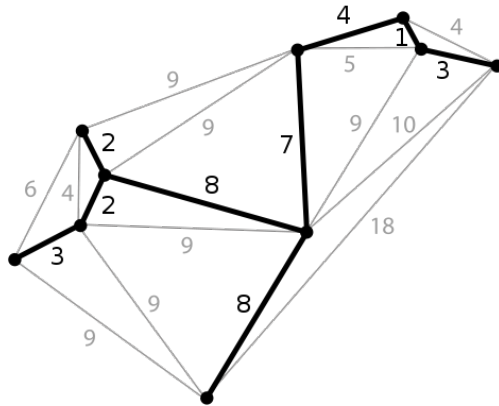


FIGURE 2.6: MINIMUM SPANNING TREE EXAMPLE (FROM [91])

if each edge has a distinct weight [90]. An MST is demonstrated in Figure 2.6.

## 2.2.2 Matrix Representation

A graph is usually stored as a list or matrix in computer systems. To understand the content of this thesis, there are some basic and important matrices to learn. One of them is called an **adjacency matrix** and is denoted by  $A$ . It is an  $n \times n$  square matrix, where  $n$  is the number of the vertices of a graph.

An unweighted adjacency matrix is defined as

$$A_{uv} = \begin{cases} 1, & \text{if } \{u, v\} \in E \\ 0, & \text{otherwise} \end{cases} \quad (2.25)$$

where 1 indicates there is an edge between node  $u$  and  $v$  whilst 0 means  $u$  and  $v$  are unconnected in a graph.

A weighted adjacency matrix is defined as

$$A_{uv} = \begin{cases} w(u, v), & \text{if } \{u, v\} \in E \\ 0, & \text{otherwise} \end{cases} \quad (2.26)$$

where  $w(u, v) \in [0, 1]$  is the weight assigned to an edge  $\{u, v\}$ .

The degree of a node  $u$ , denoted by  $deg(u)$ , is defined as

$$deg(u) = \sum_{v:\{v,u\} \in E} w(u, v) \quad (2.27)$$

For an unweighted graph, the degree of a node is the number of its neighboring nodes. The **degree matrix**, denoted by  $D$ , is defined as  $D_{ii} = deg(v_i)$  and  $D_{ij} = 0$ . In other words, the diagonal elements of  $D$  are the degrees of nodes and non-diagonal elements are set to zero.

A **Laplacian matrix**, denoted by  $L$  and also called an admittance matrix or a Kirchhoff matrix [92], is defined as the difference between a degree matrix and an adjacency matrix, i.e.  $L = D - A$ . For an unweighted graph, it is

$$L = \begin{cases} deg(u), & \text{if } u = v \\ -1, & \text{if } u \neq v \text{ and } (u, v) \in E \\ 0, & \text{otherwise} \end{cases} \quad (2.28)$$

For a weighted graph, it is

$$L = \begin{cases} deg(u), & \text{if } u = v \\ -w(u, v), & \text{if } u \neq v \text{ and } (u, v) \in E \\ 0, & \text{otherwise} \end{cases} \quad (2.29)$$

| Labeled graph | Laplacian matrix   |
|---------------|--|
|               | $\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$ |

FIGURE 2.7: GRAPH AND LAPLACIAN MATRIX REPRESENTATION (FROM [93])

An unweighted graph and its Laplacian matrix are shown in Figure 2.7.

A **distance matrix**, also called a dissimilarity matrix in the literature [94], contains the pairwise distances between data points. Depending on the definition of a distance, the matrix may not be symmetric.

## 2.3 Spectral Graph Theory

Spectral graph theory studies the properties of the *eigenvalues* and *eigenvectors* [95] of graph matrices, for example the adjacency matrix  $A$  or Laplacian matrix  $L$ .

Given an  $n \times n$  square matrix  $A$ , the characteristic equation is defined as

$$\det(A - \lambda I) = 0 \quad (2.30)$$

where  $\det$  is an operator to have the determinant of a matrix and  $I$  is the identity matrix.  $\lambda$  stands for the eigenvalues. The solutions of the characteristic equation are exactly the eigenvalues of  $A$ . The set of all eigenvalues is called the *spectrum* [96], which is invariant to the ordering of nodes. Because the spectrum of a graph is not unique, cospectral graphs may not be isomorphic [97].

By evaluating the determinant, the characteristic polynomial is expressed as a function of  $\lambda$ . The corresponding eigenvectors are defined from

$$Av = \lambda v \quad (2.31)$$



A square, symmetric and real matrix has real eigenvalues and a set of orthonormal eigenvectors. The eigen-decomposition of  $A$  can be expressed as

$$A = \Phi\Lambda\Phi^T \quad (2.32)$$

where  $\Phi$  is a square matrix whose columns are the eigenvectors of  $A$  and  $\Lambda$  is a diagonal matrix that stores the corresponding eigenvalues in its diagonal elements.

The Laplacian matrix  $L$  can also be expressed in this way. Since  $L$  is a positive semi-definite matrix [98], all of its eigenvalues are non-negative.

## 2.4 Minimum Description Length

MDL (Minimum Description Length) was first introduced by Rissanen in 1978 [99]. He believed data are maximally compressed by the best hypothesis. Based on MDL, universal codes are connected with the problems of statistical prediction [100], density estimation by stochastic complexity [101] and denoise [102]. In last few years, MDL is widely accepted as a useful mathematical tool for computer science research [103].

In principle, MDL is a trade-off theory between the goodness-of-fit and the complexity of a model [104]. The cost for coding the data by a specific hypothesis and the cost for coding the hypothesis itself are both taken into account. Mathematically, a *crude 2-part MDL* [105] is defined as

$$\min\{L_{c_1}(\mathcal{H}) + L_{c_2}(D|\mathcal{H})\} \quad (2.33)$$

where  $L_{c_1}(\mathcal{H})$  is the description length function of coding a hypothesis.  $L_{c_2}(D|\mathcal{H})$  is the description length function of coding the data with the help of a specific hypothesis. In general, description length functions are chosen according to the optimal *Shannon-Fano* code [106, 107]

$$L_{c_1}(\mathcal{H}) = -\log_2 P(\mathcal{H}) \quad (\text{bits}) \quad (2.34)$$

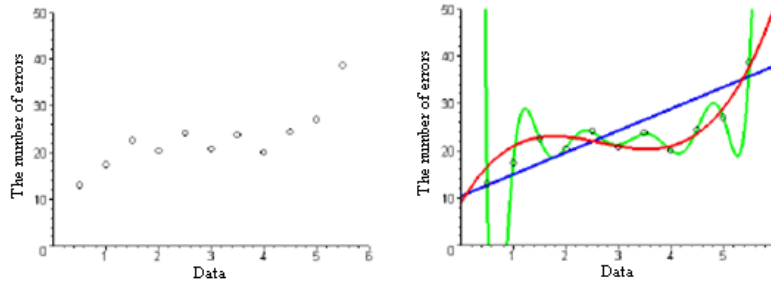


FIGURE 2.8: MDL FOR LINEAR REGRESSION PROBLEM

and

$$Lc_2(D|\mathcal{H}) = -\log_2 P(D|\mathcal{H}) \quad (\text{bits}) \quad (2.35)$$

Figure 2.8 demonstrates using MDL for a linear regression problem. As shown, the blue straight line is the simplest and cheapest hypothesis and data points are not well accommodated. The red sine function goes through most points with a small offset. The green curve can perfectly fit the data but is overly complicated and expensive to calculate. In this example, the red sine function is the optimal hypothesis. Thus it can be seen that the 2-part MDL tends to select the hypothesis which is not too simple nor too complex.

MDL was also adopted for merging clusters in a hierarchical manner [36]. As it has been reviewed, given a fixed coding scheme with its associated coding length function, an optimal clustering is the one that minimizes the total coding lengths over all possible clusters. Finding the global minimum of the overall coding lengths is a daunting combinatorial optimization problem and is often intractable for large data sets, hence an agglomerative clustering algorithm is proposed to effectively minimize the coding length in steepest descent fashion. The algorithm terminates when the coding length cannot be further reduced by merging any pair of clusters.

## 2.5 Graph Characterization and Node Selection

In this section, the research on the spectral characterization of graphs is reviewed briefly. Graph characterization methods by using the heat kernel and commute time are presented. Centralities of a graph are also introduced as the measures useful for characterizing the graph and selecting its most important nodes.

### 2.5.1 Spectral Characterization of Graphs

In the literature, the widely used characteristics of graphs are the topological properties, for example vertex set cardinality, edge density, graph perimeter and volume [108]. However topological features are of an exponentially computational complexity [109], which limits their use for graph characterization.

For the past twenty years, spectral graph theory has been applied to graph characterizing and matching problems. The same-size graph matching method was proposed by Umeyama using eigendecomposition technique [110]. The feature correspondence was established by Shapiro and Brady using the eigenvectors of a weighted pairwise distance matrix [111]. Juvan and Mohar applied the Fiedler vector of a graph Laplacian to realize spectral sequencing from ranking vector components [112]. Atkins et al. used the Fiedler vector to sequence relational data [113]. A new graph retrieval method was also proposed by Shokoufandeh et al. using an indexing mechanism which maps the topological structure of shock-trees to a low-dimensional vector space [114].

More recently, Robles-Kelly and Hancock proposed a spectral seriation method to recover a graph path with edge connectivity constraints [115]. They also defined the serial ordering of the nodes in a graph by using the leading eigenvector of a transition matrix [116]. In the meantime, Luo et al. adopted a spectral approach to learn the structural variations in graphs [117]. Qiu and Hancock de-

veloped an inexact graph matching method by using the commute time matrix and its spectral properties [118, 119]. The heat kernel was also used for graph characterizing and matching [120, 121].

## 2.5.2 Graph Characterization by Heat Kernel

The heat kernel represents the evolution of temperature in a region around a point with initial unit of heat energy at time  $t = 0$ . For a graph  $G(V, E)$ , the heat kernel depicts how information flows through edges over time. Given an unnormalized weighted Laplacian  $L$ , a normalized weighted Laplacian matrix  $\mathcal{L} = D^{-1/2}LD^{-1/2}$  is expressed concretely

$$\mathcal{L} = \begin{cases} 1, & \text{if } u = v \\ -\frac{w_{uv}}{\sqrt{d_u d_v}}, & \text{if } u \neq v \text{ and } (u, v) \in E \\ 0, & \text{otherwise} \end{cases} \quad (2.36)$$

$\mathcal{L}$  can be decomposed to  $\Phi\Lambda\Phi^T$ , where  $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_{|V|})$  is a diagonal matrix having the ordered eigenvalues  $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_{|V|} \leq 2$  [96] and  $\Phi = (\phi_1 | \phi_2 | \dots | \phi_{|V|})$  is a square matrix with the ordered eigenvectors in columns.

The heat equation associated with  $\mathcal{L}$  is given as

$$\frac{\partial H_t}{\partial t} = -\mathcal{L}H_t \quad (2.37)$$

where  $t$  is the time and the heat kernel  $H_t$  is the solution of the heat equation

$$H_t = e^{-t\mathcal{L}} \quad (2.38)$$

$H_t$  can also be expressed as

$$H_t = \Phi e^{-t\Lambda} \Phi^T \quad (2.39)$$

The element  $H_t(u, v)$  is

$$H_t(u, v) = \sum_{i=1}^{|V|} e^{-\lambda_i t} \phi_i(u) \phi_i(v) \quad (2.40)$$

A normalized adjacency matrix is defined as  $\mathcal{A} = D^{-1/2}AD^{-1/2}$ . The relationship between  $\mathcal{L}$  and  $\mathcal{A}$  is

$$\mathcal{A} = I - \mathcal{L} \quad (2.41)$$

where  $I$  is the identity matrix.

Thus  $H_t$  can be expressed as

$$H_t = e^{-t(I-\mathcal{A})} = e^{-t}e^{t\mathcal{A}} \quad (2.42)$$

The trace of  $H_t$  is

$$Tr[H_t] = \sum_{i=1}^{|V|} e^{-\lambda_i t} \quad (2.43)$$

The zeta function is defined with the eigenvalues of  $\mathcal{L}$

$$\zeta(s) = \sum_{\lambda_i \neq 0} \lambda_i^{-s} \quad (2.44)$$

The zeta function and the trace of  $H_t$  are linked via the *Mellin transform* [122]

$$\zeta(s) = \frac{1}{\Gamma(s)} \int_0^\infty t^{s-1} \left\{ Tr[H_t] - C \right\} dt \quad (2.45)$$

where  $C$  is the number of eigenvalues which are equal to zero and  $\Gamma(s)$  is the gamma function defined as

$$\Gamma(s) = \int_0^\infty t^{s-1} e^{-t} dt \quad (2.46)$$

A feature vector of the moments of the zeta function  $\mathbf{z} = (\zeta(1), \zeta(2), \dots, \zeta(k))^T$  was taken to characterize a graph, where  $k = 6$  [120].

The graph characterization algorithm is given as:

1. Compute the normalized Laplacian matrix  $\mathcal{L}$  for a graph.
2. Eigen-decompose  $\mathcal{L}$  for non-zero eigenvalues.
3. Compute the zeta function  $\zeta(s)$  using Equation (2.44).
4. Use the feature vector  $\mathbf{z}$  of a selection of  $\zeta(s)$  to characterize the graph.

### 2.5.3 Graph Characterization by Commute Time

The commute time is closely related to the *Green's function* [123]. Given the Laplace operator  $\Delta = D^{-1/2}\mathcal{L}D^{1/2}$  on a graph, where  $D$  is the degree matrix and  $\mathcal{L}$  is the normalized Laplacian, the Green's function  $G$  is the left inverse operator of  $\Delta$

$$G\Delta = I \quad (2.47)$$

The normalized Green's function is defined as  $\mathcal{G} = D^{1/2}GD^{-1/2}$ , so we have

$$\mathcal{G}\mathcal{L} = \mathcal{L}\mathcal{G} = I \quad (2.48)$$

The normalized Green's function and the heat kernel are related as

$$\mathcal{G} = \int_0^\infty H_t dt \quad (2.49)$$

For the unnormalized Green's function, the relationship between  $G(u, v)$  and  $H_t(u, v)$  is

$$G(u, v) = \int_0^\infty d_u^{1/2} H_t(u, v) d_v^{-1/2} dt \quad (2.50)$$

where  $d_u$  is the degree of node  $u$ .

Since  $\int_0^\infty e^{-t\lambda_i} dt = 1/\lambda_i$ , Equation (2.50) can be further written as

$$G(u, v) = \sum_{i=2}^{|V|} \frac{1}{\lambda_i} d_u^{1/2} \phi_i(u) \phi_i(v) d_v^{-1/2} \quad (2.51)$$

where  $\phi_i$  and  $\lambda_i$  are the  $i$ -th eigenvector and eigenvalue of  $\mathcal{L}$ .

The Green's functions with no boundaries are slightly different as

$$\mathcal{G}(u, v) = \sum_{i=2}^{|V|} \frac{1}{\lambda_i} \phi_i(u) \phi_i(v) \quad (2.52)$$

and

$$G(u, v) = \int_0^\infty d_u^{1/2} (H_t(u, v) - \phi_1(u) \phi_1(v)) d_v^{-1/2} dt \quad (2.53)$$

The hitting time  $Q(u, v)$  between nodes  $u$  and  $v$  is defined as the expected number of steps of a random walk from  $u$  to  $v$  on a graph [119]

$$Q(u, v) = \frac{vol}{d_v}G(v, v) - \frac{vol}{d_u}G(u, v) \quad (2.54)$$

where  $vol$  is the sum of all degrees in the graph.

The commute time between  $u$  and  $v$  is defined as  $CT(u, v) = Q(u, v) + Q(v, u)$  and concretely

$$CT(u, v) = \frac{vol}{d_u}G(u, u) + \frac{vol}{d_v}G(v, v) - \frac{vol}{d_u}G(u, v) - \frac{vol}{d_v}G(v, u) \quad (2.55)$$

The commute time on a graph is a metric [124]. If a pair of nodes are close or connected by many paths, the commute time between them is small. Thus a commute time matrix is also regarded as a distance matrix.

The commute time can be used to embed a graph to a Hilbert subspace via rewriting  $CT(u, v)$  as [118]

$$CT(u, v) = \sum_{i=2}^{|V|} \left( \sqrt{\frac{vol}{\lambda_i d_u}} \phi_i(u) - \sqrt{\frac{vol}{\lambda_i d_v}} \phi_i(v) \right)^2 \quad (2.56)$$

where  $\sqrt{\frac{vol}{\lambda_i d_u}} \phi_i(u)$  is regarded as the  $i$ -th coordinate of node  $u$  in the new space.

The commute time embedding algorithm is summarized as:

1. Compute the normalized Laplacian matrix  $\mathcal{L}$  for a graph.
2. Eigen-decompose  $\mathcal{L}$  to have non-zero eigenvalues and eigenvectors.
3. Compute the coordinates of nodes in the new space using Equation (2.56).

## 2.5.4 Node Selection by Centralities

In graph theory, the centrality of a vertex measures its relative importance within a graph. There are four main measures of centrality: degree, closeness, betweenness and eigenvector [125].

### **Degree Centrality**

Degree centrality is defined as the number of edges incident upon a node [126]. In the case of a directed graph, two separate measures of degree centrality are defined, namely indegree and outdegree. Accordingly, indegree is the number of edges directed to the node and outdegree is the number of edges that the node directs to others. Mathematically, the degree centrality of a node  $u$  for an undirected graph  $G = (V, E)$  with  $|V|$  vertices and  $|E|$  edges is

$$C_D(u) = \text{deg}(u) \quad (2.57)$$

### **Closeness Centrality**

In connected graphs, the distance between a pair of nodes can be defined by the length of their shortest path, which is also called a graph geodesic [127]. The farness of a node  $u$  is defined as the sum of its distances to all other nodes, and closeness centrality is defined as the inverse of the farness [128]. Thus, the more central a node is, the lower its total distance to all other nodes. Closeness centrality can be regarded as a measure of the time taken to spread information from  $u$  to all other nodes sequentially. Mathematically, the closeness centrality of a node  $u$  is defined as

$$C_C(u) = \frac{1}{\sum_{t \in V} d_G(u, t)} \quad (2.58)$$

where  $d_G(u, t)$  denotes the graph geodesic from node  $u$  to  $t$ .

### **Betweenness Centrality**

Betweenness is a centrality measure of a node within a graph which quantifies the number of times the node acts as a bridge along the shortest path between two other nodes [129]. Hence, nodes that have a high probability to occur on a



randomly chosen shortest path between two randomly chosen nodes have a high betweenness. The mathematical expression of betweenness centrality is

$$C_B(u) = \sum_{(s,t) \in V} \frac{\sigma_{st}(u)}{\sigma_{st}} \quad (2.59)$$

where  $\sigma_{st}$  is the total number of shortest paths from node  $s$  to  $t$  and  $\sigma_{st}(u)$  is the number of those paths that pass through  $u$ .

### Eigenvector Centrality

Eigenvector centrality was proposed to measure the influence of a node in a graph [130]. Relative scores are assigned to all nodes in the graph based on the concept that for the node in question, connections to high-scoring nodes contribute more than equal connections to low-scoring nodes. Google's PageRank is a variant of the eigenvector centrality measure [131]. Given the graph  $G$  and its adjacency matrix  $A$ , the eigenvector centrality score of a node  $u$  is defined as

$$C_E(u) = \frac{1}{\lambda} \sum_{v \in G} A_{uv} C_E(v) \quad (2.60)$$

where  $\lambda$  is a constant. Rewrite (2.60) in vector notation with  $x$  replacing  $C_E$

$$Ax = \lambda x \quad (2.61)$$

In general, there are many different eigenvalues  $\lambda$  for which an eigenvector solution exists. However, as it is additionally required that all the entries in the eigenvector are positive, only the greatest eigenvalue results in the desired centrality measure [132]. The  $u^{th}$  component of the eigenvector gives the centrality score of node  $u$  in the graph.

## 2.6 Feature Detecting and Image Matching

This section, although peripheral to the main topic of the graph-based image matching chapter, briefly introduces two image matching techniques which are used in our method in the hope that the thesis can be more self-contained.

### 2.6.1 Harris Corner Detecting

An image is matched by its feature points, which can be visually meaningful, e.g. edges [133] and corners [134], or more abstractly defined, e.g. blobs [135] and ridges [12]. A corner is regarded as the point with low self-similarity by an early corner detection algorithm Moravec [136]. Harris and Stephens improved Moravec by adopting the differential of a corner score with respect to directions [137].

Given a two dimensional grayscale image  $I$ , an image patch is taken over the area  $(u, v)$  and shifted by  $(x, y)$  pixels, the weighted SSD (Sum of Squared Differences) between these two patches is defined as

$$S(x, y) = \sum_u \sum_v w(u, v) (I(u + x, v + y) - I(u, v))^2 \quad (2.62)$$

where it is better for  $w(u, v)$  to be a smooth circular window rather than a square patch so that it will be insensitive to an in-plane image rotation. A Gaussian filter is such a perfect smooth circular window

$$w(u, v) = e^{-(u^2+v^2)/2\sigma^2} \quad (2.63)$$

When performing corner detection in practice, the information of local features are missing, so we can only compute the difference with respect to small variations in position, i.e.  $(x \ y) = (\Delta u \ \Delta v)$ . As a result, Equation (2.62) is transformed to an *auto-correlation function* [138].

Let  $I_x$  and  $I_y$  denote the partial derivatives of  $I$ , via the *Taylor expansion* [139]

$$I(u + x, v + y) \approx I(u, v) + I_x(u, v)x + I_y(u, v)y \quad (2.64)$$

Hence  $S(x, y)$  can be approximated as

$$S(x, y) \approx \sum_u \sum_v w(u, v) (I_x(u, v)x + I_y(u, v)y)^2 \quad (2.65)$$

In matrix form

$$S(x, y) \approx (x \ y) A (x \ y)^T \quad (2.66)$$

where  $A$  is called the Harris matrix [140]

$$A = \sum_u \sum_v w(u, v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (2.67)$$

A corner is characterized as causing a large variation of  $S$  in all directions of  $(x \ y)$ . This is equally interpreted as that if  $A$  has two large positive eigenvalues, the area  $(u, v)$  must contain a corner point.

To measure the quality of corners for selection, a response function is defined

$$R = \det(A) - k[\text{Tr}(A)]^2 = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \quad (2.68)$$

where  $R$  is positive for corners and negative for edges.  $k$  is set empirically in the range of [0.04 0.15] for the best performance. To express  $\lambda_1 \gg 0$  and  $\lambda_2 \gg 0$  by  $R$ , we can define  $R = \min(\lambda_1, \lambda_2)$  [134] or  $R = \frac{\det(A)}{\text{Tr}(A)} = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$  [141].

There is a problem with this method as sometimes in regions of higher contrast, there are too many corners than other areas. To solve it, the *non-maximal suppression* [142] is proposed to only keep the corners which are local maxima and have  $R$  value significantly greater than their  $\epsilon$ -neighbors, where  $\epsilon$  is a radius. In this way, corners are uniformly distributed across an image.

## 2.6.2 SIFT Image Matching

SIFT (Scale-Invariant Feature Transform) image matching method was proposed by Lowe in 1999 [14]. This method was soon acknowledged as one of the most effective and successful mainstream matching methods. The main steps of the method to detect and describe image features can be summarized as [143]:

1. Scale-space extrema detection. At this stage the potential interest points are identified by using a DoG (Difference-of-Gaussian) function, which are invariant to scale and orientation.
2. Keypoint localization. At this stage keypoints are selected from candidate points based on a stability measure.
3. Orientation assignment. At this stage local image gradient orientations are assigned to each keypoint.
4. Keypoint descriptor. At this stage local image gradients are measured at selected scales in the region around each keypoint. A vector representation of the descriptor is built to allow for significant levels of local shape distortion and change in illumination.

The SIFT keypoint descriptor is inspired by a biological vision study [144]. A SIFT descriptor is illustrated in Figure 2.9. On the left, the gradient magnitude and orientation at each image pixel in the region around a keypoint are computed and then weighted by a Gaussian window (the blue overlaid circle). On the right, the contents of accumulated orientation histograms are summarized over  $4 \times 4$  subregions. There are 8 orientation bins and the length of each arrow corresponds to the sum of the gradient magnitudes near that direction.

A SIFT image matching is demonstrated in Figure 2.10. The left and right images have 874 and 849 keypoints respectively. Totally there are 402 matches

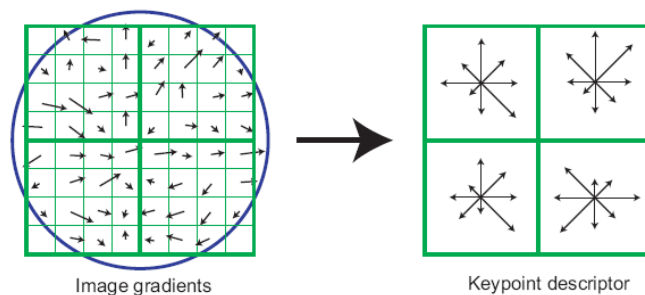


FIGURE 2.9: SIFT DESCRIPTOR REPRESENTATION (FROM [143])



FIGURE 2.10: SIFT IMAGE MATCHING EXAMPLE

found and only a small portion is shown here. The matching accuracy in this example is 99.99%.

## 2.7 Summary

In this chapter, the research literature on data clustering is reviewed. The background knowledge and clustering methods related to our work are introduced. It is noticeable that most current clustering methods are not capable of dealing with

different data types. Many of them require a set of internal parameters to work properly. However, the parameter tuning process is often experiment-based and sometimes complicated. Noise can also affect the clustering performance significantly. These problems should be addressed for better cluster analysis.

The literature on spectral characterization of graphs is also briefly reviewed. The graph characterization and node selection methods using heat kernel, commute time and centrality measures are presented. Two popular techniques for image matching, namely the Harris corner detector and SIFT descriptor, are also introduced since they are adopted in our image matching method.

## Chapter 3

# M1NN Agglomerative Clustering

In this chapter, we present a novel agglomerative clustering method based on the M1NN principle and parallel clustering algorithm. The M1NN principle emphasizes that a pair of nodes in a mutual 1-NN graph forms the basic unit of a cluster. This method is robust to outliers and could be adapted for data mining, image segmentation and manifold learning tasks.

### 3.1 M1NN Principle

What is a cluster? This is a fundamentally important question for cluster analysis. Many clustering methods have their own definitions. For example, K-means takes clusters as non-intersected convex balls [8] whilst DBSCAN picks them from spatially denser regions [51]. Clusters are also defined as the subgraphs called maximal cliques [145] or dominant sets [61] by the clustering methods based on graph theory. Because there is not a universally and mathematically precise description of clusters, and designing a general algorithm for different types of data is particularly difficult, clustering is considered as an ill-posed and challenging problem [146].

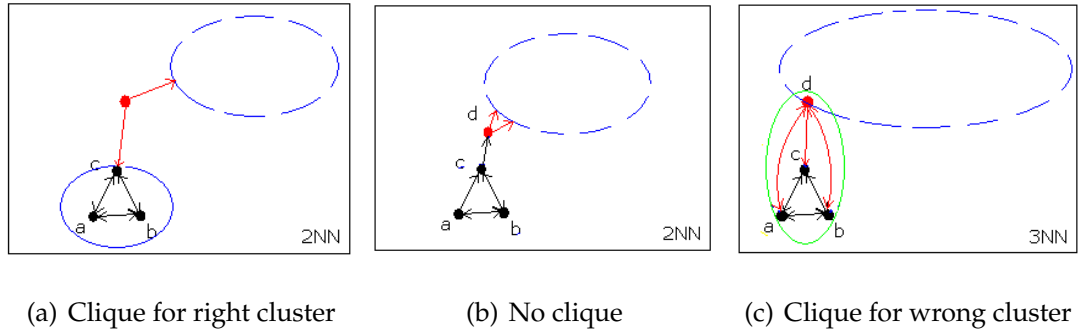


FIGURE 3.1: OBSERVATION 2 DEMONSTRATION

Essentially, there are two criteria for defining a cluster. The internal criterion requires all the objects inside a cluster are highly similar. The external criterion demands all the objects outside a cluster are highly dissimilar to the objects inside the cluster [147]. These criteria highlight the importance of the similarity between members in a cluster, which is observed and understood as follows.

First, two mutually nearest objects are most likely from the same cluster. This pair of objects can be treated as an indicator of a cluster. In the literature, a clustering method was developed on a mutual  $k$ -NN graph [148]. For multivariate mixed data, this may not be true as these two nearest objects could come from different Gaussian distributions (clusters) [149].

Second, for a set of  $n$  points and its  $k$ -NN graph, the clique of size  $k + 1$  with bi-directional edges (i.e.  $k + 1$  mutually nearest points) may not be found in the graph when  $1 < k < n - 1$ . Even there is such a clique, it may not indicate a cluster properly. In Figure 3.1, 2-NN and 3-NN graphs are plotted for a data set in which  $\{a, b, c\}$  is a cluster. Figure 3.1(a) shows a clique of size 3 corresponding to cluster  $\{a, b, c\}$ , which is unlikely to be found in practice. More often as shown in Figure 3.1(b), there is no 3-mutually-nearest-node structure to represent  $\{a, b, c\}$  since  $a$  is not a nearest neighbor for  $c$ . In Figure 3.1(c), as  $d$  may be the nearest neighbor of  $\{a, b, c\}$ , clique  $\{a, b, c, d\}$  can be misperceived as a cluster.



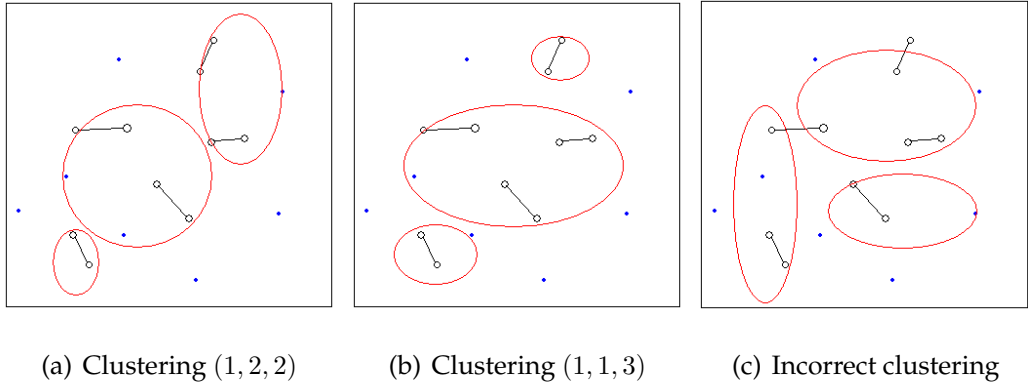


FIGURE 3.2: M1NN PRINCIPLE FOR CLUSTERING DEMONSTRATION

Thus it seems only the mutual 1-NN graph can indicate clusters accurately by a number of M1NN pairs, and each cluster should have at least one such pair. This is called the *M1NN principle* and the pair is termed a *CP* (abbr. for couple). To ensure the M1NN principle works properly, metrics should be used as non-symmetric distances may result circles in the data and there could be no CPs. Singleton clusters are excluded by the M1NN principle as a cluster should have at least two objects.

To demonstrate how to apply the M1NN principle to clustering, we assume there are 5 CPs and 3 clusters in a data set. The possible clusterings (1, 2, 2) and (1, 1, 3) are shown in Figure 3.2, in which numbers indicate how many CPs in each cluster. The CPs are denoted by the linked small black circles. The clusters are denoted by the big red circles. An incorrect clustering is shown in Figure 3.2(c) that a broken CP is assigned to two different clusters, which is against the M1NN principle.

CPs in a complex data set are demonstrated in Figure 3.3. The data set consists of 303 objects and 96 CPs. Two denser areas (clusters) are magnified to show the CPs inside. Interestingly, some of the CPs outside are even bigger than these two clusters.

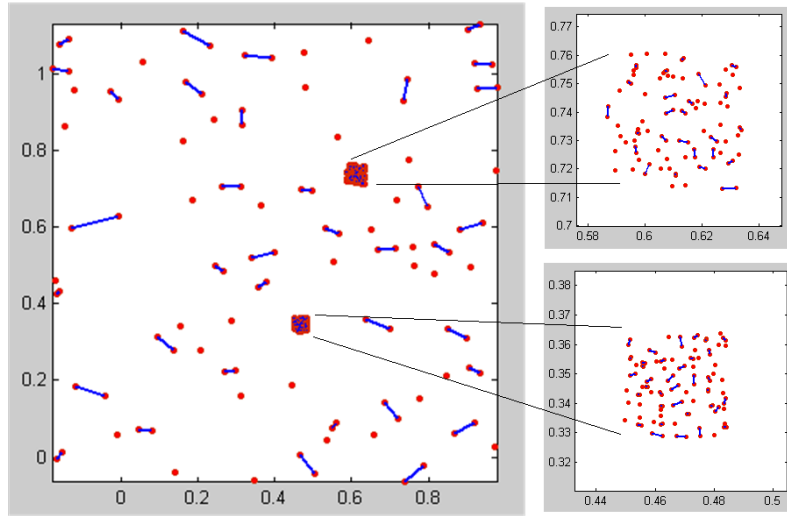


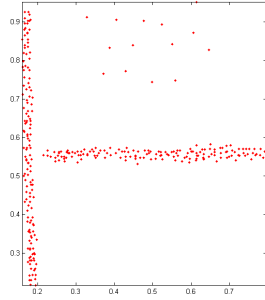
FIGURE 3.3: CPs IN A COMPLEX DATA SET

## 3.2 Parallel Agglomerative Clustering

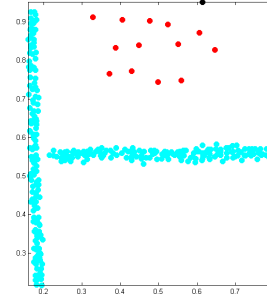
The classical single-link agglomerative clustering algorithm is 'single threaded'. Based on the M1NN principle, a parallel agglomerative clustering algorithm is proposed for improved performance.

### 3.2.1 Classical Single-Link Agglomerative Clustering

The classical single-link agglomerative clustering algorithm merges two clusters with the minimum single-linkage in each iteration until all objects are grouped in one cluster. This process is visualized by a dendrogram. To have  $k$  clusters, we can cut the dendrogram for corresponding branches. However, apparently this strategy may cause main clusters being merged earlier than supposed because of outliers and density distributions. For example, a toy data set is clustered in Figure 3.4. The sparse area negatively impacts the clustering that two long clusters are merged together at an early stage.



(a) Toy data set



(b) Classical agglomerative clustering

FIGURE 3.4: THE PROBLEM OF THE CLASSICAL AGGLOMERATIVE CLUSTERING

### 3.2.2 Parallel Clustering with M1NN Principle

The parallel clustering algorithm inherits single-linkage as the distance measure between clusters from the classical algorithm whilst changes the order of merging by adopting the M1NN principle. In Figure 3.5, these two different algorithms are compared. There are three clusters  $\{a, b, c\}$ ,  $\{d, e, f\}$  and  $\{g, h, i\}$  denoted by blue, green and red nodes in Figure 3.5(a). The classical clustering algorithm is demonstrated in Figure 3.5(b). As shown,  $a$  and  $b$  are merged at  $t = 1$  then  $d$  and  $e$  at  $t = 2$ . Next  $c$  is added to  $\{a, b\}$  and  $f$  to  $\{d, e\}$ . When  $t = 5$ , cluster  $\{a, b, c\}$  and  $\{d, e, f\}$  are grouped together. In following steps,  $g$  and  $h$  are merged into a cluster and  $i$  is added to the cluster. In the end when  $t = 8$ , all nodes are in one cluster. As mentioned before, there is a serious problem with the classical algorithm. When the dendrogram is cut at  $t = 6$  for three clusters, the result is a large one  $\{a, b, c, d, e, f\}$ , a small one  $\{g, h\}$  and a trivial one  $\{i\}$ .

On the other hand, the parallel algorithm identifies all CPs  $\{a, b\}$ ,  $\{d, e\}$  and  $\{g, h\}$  simultaneously in the first round as shown in Figure 3.5(c). Nodes not in CPs, i.e.  $c, f$  and  $i$ , are called *free nodes* and denoted by  $N_f$ . When  $t = 2$ , all the CPs are expanded to merge the free nodes nearby. For instance, node  $c$  is added to  $\{a, b\}$  and  $f$  to  $\{d, e\}$ . An expanded CP is called a CQ, which are  $\{a, b, c\}$ ,  $\{d, e, f\}$

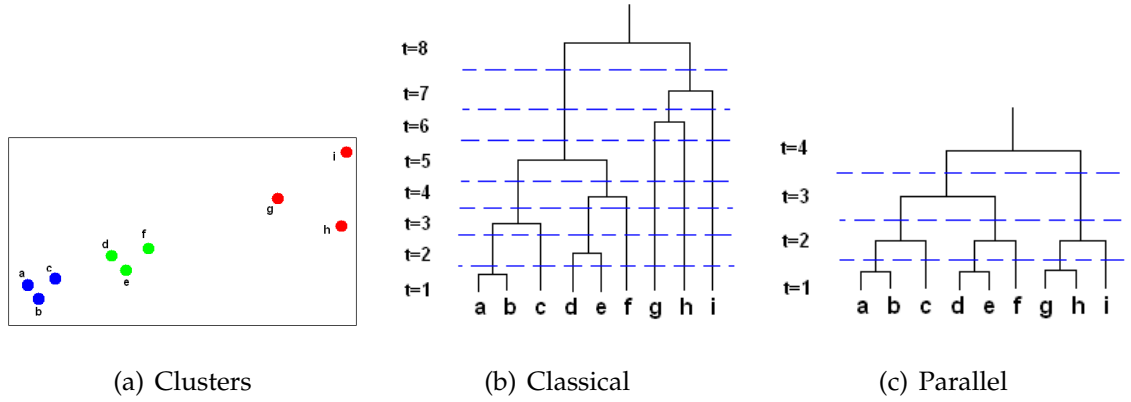


FIGURE 3.5: DENDROGRAM COMPARISON OF TWO CLUSTERING METHODS

and  $\{g, h, i\}$  in this example. The expansion of a CQ continues until any two CQs are adjacent to each other. Then CQs start to merge one another (and neighboring free nodes if any). In Figure 3.5(c),  $\{a, b, c\}$  and  $\{d, e, f\}$  are merged at  $t = 3$  and there comes only one cluster at  $t = 4$ . Again when the dendrogram is cut at  $t = 2$  for three clusters, this time the result is correct. This comparison demonstrates the advantage of using the parallel algorithm with the M1NN principle instead of using the classical algorithm for clustering analysis.

### 3.2.3 MST-Based Parallel Clustering

The classical single-link algorithm is equivalent to the Kruskal's algorithm [150], which is used for finding an MST [151]. Therefore we can retrieve the single-link clusters from an MST easily [152].

Given  $n$  data points, the single-linkage set is defined as  $SL = \{sl_1, sl_2, \dots, sl_{n-1}\}$ , where the element  $sl_i$  is the minimum weight connecting two clusters in the  $i$ -th iteration of merging and  $sl_i < sl_{i+1}$ . For simplicity, we assume the MST of the data is unique. Then we have  $SL = E_{\text{MST}}$ , where  $E_{\text{MST}}$  denotes the edge set of the MST sorted in ascending order. Hence the classical algorithm is realized by

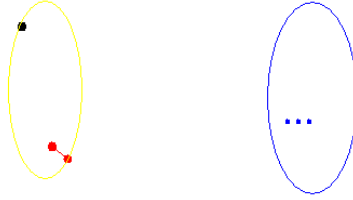


FIGURE 3.6: IMPROPER MERGING FROM PARALLEL CLUSTERING

optimally merging the data points following the order of  $E_{\text{MST}}$  along the branches of the MST.

On the other hand, the parallel algorithm is also based on MST optimization but the edge sequencing follows the M1NN principle. The clustering process can be pictured as that the CPs grow along the branches simultaneously, meeting and merging each other, forming bigger clusters until being terminated.

In practice, because of outliers, some mergings should not be allowed. For example, in Figure 3.6, a radical change will be introduced to the cluster structure if the black point is merged to the red CP. To solve this problem, the parallel algorithm is modified by adding a cluster characterizing quantity derived from the path-based dissimilarity measure.

### 3.3 Path-Based Dissimilarity Measure

The path-based dissimilarity measure was proposed by Fischer and Buhmann [153]. The idea is if data points  $i$  and  $j$  are far from each other, but there is a path connecting them and other points such that the distances between any two successive points are small, then the effective distance between  $i$  and  $j$  should be adjusted to a smaller value to reflect this connectedness. The *density reachability* [51] is actually quantified by the path-based dissimilarity measure.

To describe the path-based dissimilarity measure more formally, we assume a data set containing  $n$  objects is given and the pairwise distances are stored in an  $n \times n$  matrix  $D$ . Cluster labels are denoted by  $\Omega = \{w_1, w_2, \dots, w_k\}$ .  $s_{iw}$  is a cluster membership indicator.  $s_{iw} = 1$  means object  $i$  is in cluster  $w$ , otherwise  $s_{iw} = 0$ .  $s_w = (s_{1w}, s_{2w}, \dots, s_{nw})^T$  is the vector of indicator variables for cluster  $w$ .  $S = (s_1|s_2|\dots|s_k)$  is the cluster membership matrix.

The path-based effective dissimilarity between  $i$  and  $j$  is defined as

$$D_{ij}^{\text{eff}}(S, D) = \min_{p \in P_{ij}(S)} \left\{ \max_{h \in \{1, \dots, |p|-1\}} \{D_{p[h]p[h+1]}\} \right\} \quad (3.1)$$

where  $p$  is a path and  $h \in \{1, \dots, |p|-1\}$  denotes the labels of nodes on the path.  $D_{p[h]p[h+1]}$  is the length of the path segment between node  $p[h]$  and  $p[h+1]$ .  $P_{ij}(S)$  is the set of all the paths from  $i$  to  $j$ , defined as

$$P_{ij}(S) = \left\{ p \in \{1, \dots, n\}^l \mid \exists w : \prod_{h=1}^l s_{p[h]w} = 1 \wedge l \leq n \wedge p[1] = i \wedge p[l] = j \right\} \quad (3.2)$$

To better understand the effective dissimilarity, we assume  $i$  and  $j$  are from the same cluster and they are connected by two paths  $p_1$  and  $p_2$ . On  $p_1$ , we assume the segment between node  $a$  and  $b$  is the longest segment along the path, i.e.  $D_{p_1[a]p_1[b]} = \max_{h \in \{1, \dots, |p_1|-1\}} \{D_{p_1[h]p_1[h+1]}\}$ . Similarly on  $p_2$ , we have  $D_{p_2[m]p_2[n]} = \max_{h \in \{1, \dots, |p_2|-1\}} \{D_{p_2[h]p_2[h+1]}\}$ . The effective dissimilarity between  $i$  and  $j$  is given as  $D_{ij}^{\text{eff}}(S, D) = \min_{p_1, p_2} \{D_{p_1[a]p_1[b]}, D_{p_2[m]p_2[n]}\}$  by the definition (3.1).

More generally, when  $i$  and  $j$  are connected by multiple paths, the longest segments of each path are compared. The effective dissimilarity corresponds to the length of the shortest. For all the data points, the effective dissimilarity matrix is denoted by  $D^{\text{eff}}(S, D)$  and  $D_{ij}^{\text{eff}}(S, D)$  is its  $(i, j)$ -element.

It was believed computing the path-based dissimilarity matrix  $D^{\text{eff}}(S, D)$  is equivalent to solving an *all-pairs-shortest-path* problem [154]. Here we prove that  $D^{\text{eff}}(S, D)$  is actually obtained by solving an MST problem (for the difference, please refer to [155]). Thus the computational complexity is greatly reduced.

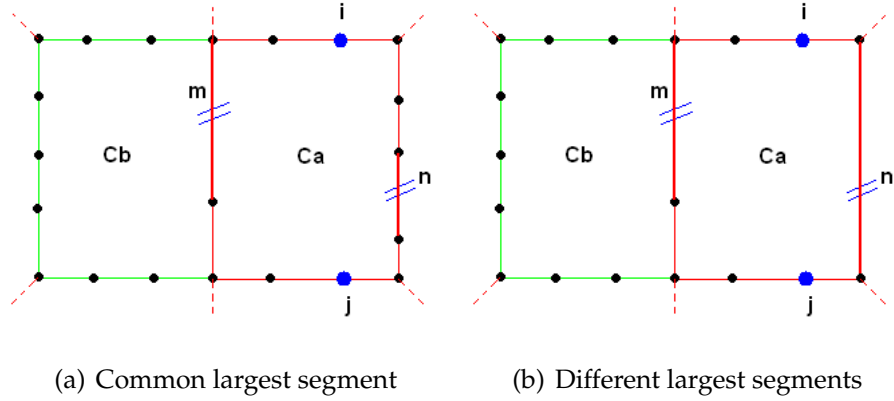


FIGURE 3.7: SEGMENT REMOVAL DEMONSTRATION

Generally for two data points  $i$  and  $j$  in a cluster, the set of paths connecting them is denoted by  $P_{ij} = \{p_1, p_2, \dots, p_k\}$ . First we consider a simple situation that there are only two paths from  $i$  to  $j$ , i.e.  $k = 2$ . A cycle is formed between  $p_1$  and  $p_2$ .  $D_{p_1[a]p_1[b]}$  and  $D_{p_2[m]p_2[n]}$  still denote the largest segments of each path. If we assume  $D_{ij}^{\text{eff}}(S, D) = D_{p_1[a]p_1[b]}$ , then  $D_{p_2[m]p_2[n]}$  can be safely removed from the cycle without affecting the computation of effective dissimilarities between the nodes on the cycle.

Similarly when  $i$  and  $j$  are connected by multiple paths, there are many more cycles. On each cycle, the largest segment can be safely dropped. This is because even if the dropped segment is not the largest on another cycle, for the nodes on that cycle there must be a path on which the largest segment is smaller than this one. In Figure 3.7, the procedure of segment removal is demonstrated. In Figure 3.7(a), edge  $m$  is the common largest segment of cycle  $C_a$  and  $C_b$ . Upon removing edge  $m$ , edge  $n$  becomes the largest segment of the joint cycle and is removed as well. In Figure 3.7(b), edge  $m$  is the largest segment for  $C_b$  only. However its deletion does not affect either removing edge  $n$  or computing the effective dissimilarities between the nodes on  $C_a$ . Clearly for node  $i$  and  $j$ , in both cases, their effective dissimilarity is decided by the longest path along  $C_a$  and  $C_b$ .

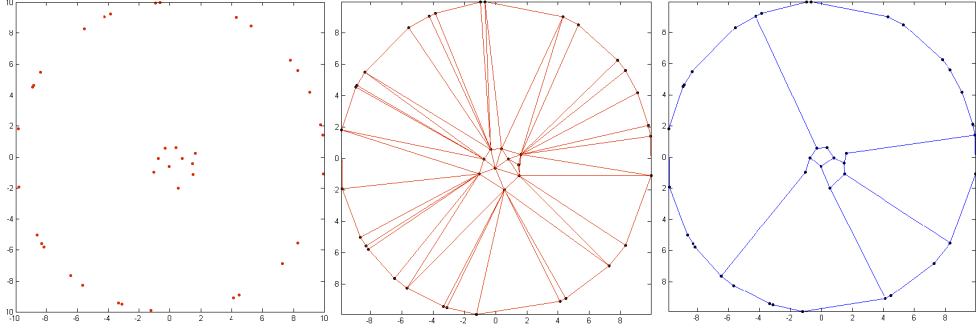


FIGURE 3.8: DELAUNAY GRAPH AFTER REMOVING TRIANGLES

Further for all the points, if all the cycles are broken on their largest segments, there remains a tree. The tree is exactly an MST according to the cycle property: for any cycle in a graph, if the weight of an edge of the cycle is larger than the weights of other edges of the cycle, this edge cannot belong to an MST [156].

Figure 3.8 demonstrates a weighted Delaunay graph with the largest edges of all the triangles removed. For any two nodes  $i$  and  $j$  on the simpler graph,  $D_{ij}^{\text{eff}}(S, D)$  does not change. When all the cycles are broken, an MST will appear in the end. The effective dissimilarity  $D_{ij}^{\text{eff}}(S, D)$  can be computed from the largest segment of the path from  $i$  to  $j$  on the MST.

For a cluster of data points, the characterizing quantity  $\kappa$  is defined as

$$\kappa(w) = \max_{\{i,j\} \in w} \left\{ D_{ij}^{\text{eff}}(S, D) \right\} \quad (3.3)$$

where  $i$  and  $j$  denote the objects in cluster  $w$ .

Since we have proved the path-based effective dissimilarity can be obtained from an MST,  $\kappa$  as the maximum dissimilarity, is equal to the *bottleneck edge* of the MST. In Figure 3.9, the clusters are shown in the first row. The corresponding MSTs are plotted in the second row. The bottleneck edges are highlighted in green color.



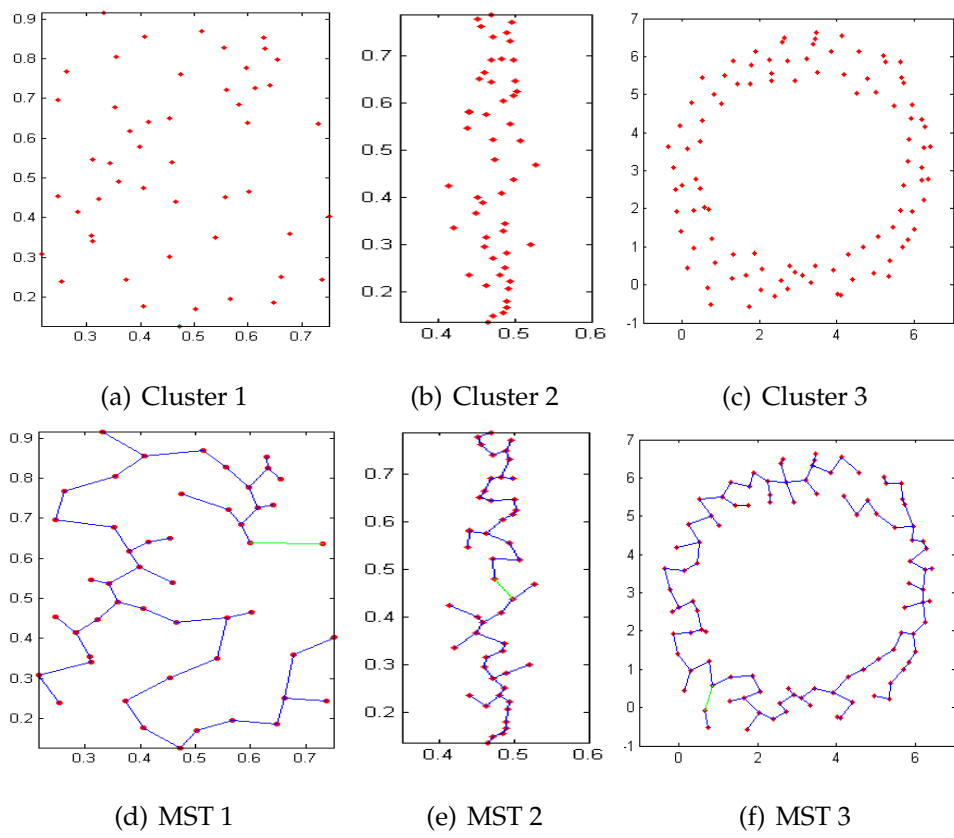


FIGURE 3.9: THE BOTTLENECK EDGES DEMONSTRATION

## 3.4 M1NN Agglomerative Clustering

The M1NN agglomerative clustering method is based on the parallel clustering algorithm with the cluster characterizing quantity  $\kappa$  as a structural constraint. There are three main steps which are described explicitly in each section below.

### 3.4.1 CP to CQ Expansion

Given a cluster  $w$  and a node  $i$  from  $N_f$ , the distance between them is defined in a similar way to single-linkage

$$d(i, w) = \min_{j \in w} \left\{ d_{ij} \right\} \quad (3.4)$$

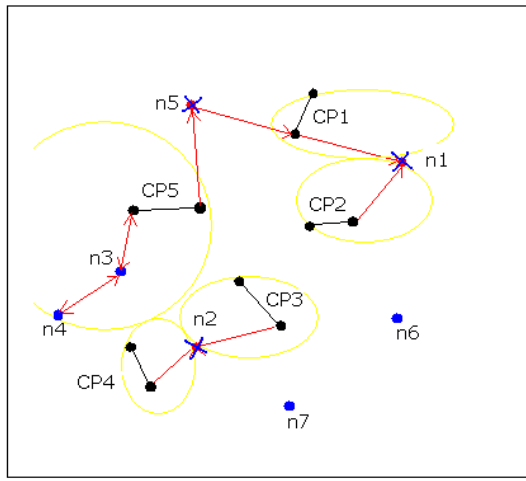
where  $j$  is a node in cluster  $w$  and  $d$  denotes a metric measure, e.g. Euclidean distance in our method.

The CP to CQ expansion is guided by the M1NN principle and  $\kappa$  condition. If we assume node  $p$  and  $CP_i$  are mutually nearest and  $p$  is not the nearest node for another  $CP_j$ , i.e.  $(CP_i \leftrightarrow p \nleftrightarrow CP_j)$ , then  $p$  is merged to  $CP_i$  if it satisfies

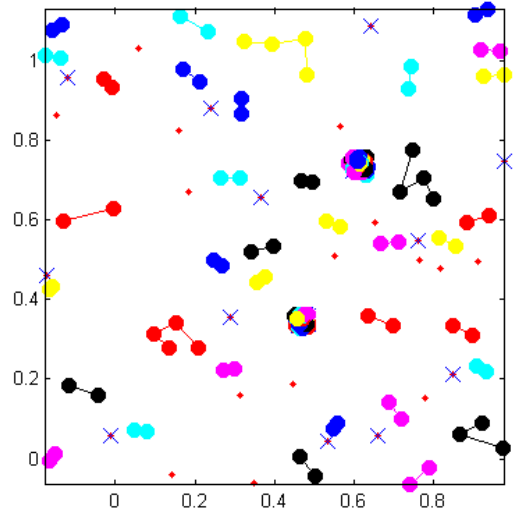
$$\alpha * \kappa(CP_i) \geq d(p, CP_i) \quad (3.5)$$

where  $\alpha$  is a parameter to control the expansion. For example in Figure 3.6, the red CP will not merge the black node if  $\alpha < 5$ . In our experiments,  $\alpha$  is set to 2.

On the other hand, if we assume  $p$  is the nearest node for both  $CP_i$  and  $CP_j$ , or  $CP_i$  is nearest to  $p$  but  $p$  is nearest to  $CP_j$ , i.e.  $(CP_i \rightarrow p \leftarrow CP_j)$  or  $(CP_i \rightarrow p \rightarrow CP_j \rightarrow q)$  where  $q$  denotes another node, then  $p$  is defined as a boundary node and cannot be merged to the CPs. The expansions of  $CP_i$  and  $CP_j$  are then terminated. Essentially we aim to expand all the CPs to have as many small and adjacent clusters as possible in this step so that the single-linkage can capture the distances between these clusters more accurately.



(a) CP to CQ expansion



(b) CP expansion on a complex data set

FIGURE 3.10: CP TO CQ EXPANSION

The expansion is demonstrated in Figure 3.10(a) when the  $\kappa$  condition is assumed to be satisfied with a proper  $\alpha$  value for all the CPs. As shown,  $CP_1$  and  $CP_2$  have a common nearest node  $n_1$ , therefore they are not expanded and  $n_1$  is taken as a boundary node.  $CP_3$  and  $CP_4$  are not expanded as well because of the boundary node  $n_2$ .  $CP_5$  is expanded to merge  $n_3$  and  $n_4$ , then the expansion is stopped because  $n_5$  as the next nearest node for  $CP_5$  is itself nearest to  $CP_1$ . Figure 3.10(b) shows CP expansion on the complex data set from Figure 3.3. Crossed nodes denote the boundary nodes. Unmarked nodes are not visited yet.

### 3.4.2 CQ Merging Rules

The CQ merging process corresponds to the steps from  $t \geq 3$  in Figure 3.5(c). Similar to CP expansion, CQs are merged following the M1NN principle and  $\kappa$  condition. If we assume a CQ set is given as  $\Omega = \{CQ_1, CQ_2, \dots, CQ_k\}$ , in which

$CQ_i$  and  $CQ_j$  are two mutually nearest CQs, we have

$$\begin{cases} sl(CQ_i, CQ_j) < \{sl(CQ_i, CQ_u) | \forall CQ_u \in \Omega \wedge u \neq j\} \\ sl(CQ_j, CQ_i) < \{sl(CQ_j, CQ_v) | \forall CQ_v \in \Omega \wedge v \neq i\} \end{cases} \quad (3.6)$$

where  $sl$  is the single-linkage between two CQs. In the meantime if the  $\kappa$  condition is satisfied

$$\begin{cases} \beta * \kappa(CQ_i) \geq sl(CQ_i, CQ_j) \\ \beta * \kappa(CQ_j) \geq sl(CQ_i, CQ_j) \end{cases} \quad (3.7)$$

where  $\beta$  is a parameter similar to  $\alpha$  (we also set  $\beta = 2$  in our experiments), then  $CQ_i$  and  $CQ_j$  will be merged and the new cluster is called an MCQ. Given all  $m$  new clusters generated from merging mutually nearest CQs at one iteration,  $\Omega_m = \{MCQ_1, MCQ_2, \dots, MCQ_m\}$  is defined as the set of all MCQs. A remaining cluster which is not merged at the iteration is called a RCQ. For all  $n$  remaining clusters,  $\Omega_r = \{RCQ_1, RCQ_2, \dots, RCQ_n\}$  is defined as the set of all RCQs.

Once an MCQ is generated, it is updated by merging free nodes nearby. There are two update rules for the nodes 'inside' and 'outside' an MCQ. Given  $p$  as the nearest node for  $MCQ_i$ , it will be merged to  $MCQ_i$  if it satisfies

$$\kappa(MCQ_i) > \kappa(\{p, MCQ_i\}) \quad (3.8)$$

which indicates  $p$  is inside  $MCQ_i$ . Otherwise the  $\kappa$  condition applies to decide if  $p$  is merged to  $MCQ_i$

$$\alpha * \kappa(MCQ_i) \geq d(p, MCQ_i) \quad (3.9)$$

The CQ set is updated as  $\Omega = \{\Omega_m \cup \Omega_r\}$ . This merging process repeats until

$$|\Omega_m^{(x)}| < k \quad (3.10)$$

where  $k$  is the user-defined cluster number and  $\Omega_m^{(x)}$  is the MCQ set on  $x$ -th iteration. The final cluster number is  $|\Omega_m^{(x-1)}| + |\Omega_r^{(x-1)}|$ , which is usually bigger than  $k$ .

The expression (3.10) is used as a terminating condition as we assume the main clusters are contained in the final MCQ set. Ideally the MCQs will be the true clusters. The worst case is that all the MCQs are from one cluster which could result 'under' clustering.

Inspired by DBSCAN, the parameter *MinPts* is adopted to set a size limit to clusters. Any final clusters smaller than *MinPts* are broken and their nodes are freed for other clusters to merge. The pseudocode of the CQ merging algorithm is shown in Figure 3.11. The CQ merging result for Figure 3.10(b) is shown in Figure 3.12 where *MinPts* = 2.

### 3.4.3 M1NN Agglomerative Clustering with Strong Merging Rule

The clustering from CQ merging is a little sparse as shown in Figure 3.12. This 'under' clustering is appropriate when accurate cluster types or numbers are unknown, in which case the main clusters are captured with minimum errors based on the MST optimization and M1NN principle.

However sometimes, the exact number of clusters is expected. A strong merging rule is proposed to further process the clustering from CQ merging step until there are approximately  $k$  clusters. The algorithm is realized by using the classical algorithm with a compatibility function.

Given a CQ set, each CQ is taken as a node and a  $k$ -NN graph is constructed to connect all the CQs where  $k$  is the cluster number and  $\mathcal{N}(CQ_i)$  denotes neighboring CQs of  $CQ_i$ . The compatibility function between  $CQ_i$  and  $CQ_j$  is defined as

$$R(CQ_i, CQ_j) = \max_{CQ_j \in \mathcal{N}(CQ_i)} \left\{ \frac{sl(CQ_i, CQ_j)}{\kappa(CQ_i)}, \frac{sl(CQ_i, CQ_j)}{\kappa(CQ_j)} \right\} \quad (3.11)$$

where  $sl(CQ_i, CQ_j)$  is the single-linkage between  $CQ_i$  and  $CQ_j$ , which is larger than either  $\kappa(CQ_i)$  or  $\kappa(CQ_j)$ . This function emphasizes the impact of the cluster

---

**Algorithm 1** CQ merging algorithm

---

**Require:** user-defined cluster number  $k$   
the minimum size of a cluster  $MinPts$   
Euclidean distance function between data points  $d(.,.)$   
single linkage between clusters based on distance function  $sl(.,.)$   
cluster function to merge clusters or nodes to one cluster  $merge()$   
cluster function to return how many objects in a cluster  $size()$   
cluster function to release all objects in a cluster  $break()$   
initial  $CQ^{(0)}$  set and free nodes set  $N_f^{(0)}$   
initialize  $\Omega_m = CQ^{(0)}$ ,  $\Omega_r = \emptyset$  and  $x = 0$

- 1: **while**  $|\Omega_m| \geq k$  **do**
- 2:      $x = x + 1$
- 3:     **for all**  $(i, j) \in \{1, \dots, |CQ|\}$  **do**
- 4:         **if**  $sl(CQ_i, CQ_j) < sl(CQ_i, CQ_a) \wedge sl(CQ_i, CQ_j) < sl(CQ_j, CQ_b) \wedge$   
        $\{a, b\} \neq \{i, j\} \wedge \beta * \kappa(CQ_i) \geq sl(CQ_i, CQ_j) \wedge \beta * \kappa(CQ_j) \geq sl(CQ_i, CQ_j)$   
       **then**
- 5:              $merge(CQ_i, CQ_j)$  and save to  $\Omega_m^{(x)}$
- 6:             **end if**
- 7:     **end for**
- 8:     **for all**  $m \in \{1, \dots, |\Omega_m|\} \wedge n \in \{1, \dots, |N_f|\}$  **do**
- 9:         **if**  $\kappa(MCQ_m) > \kappa(\{N_{f_n}, MCQ_m\}) \vee \alpha * \kappa(MCQ_m) \geq$   
        $d(N_{f_n}, MCQ_m)$  **then**
- 10:              $merge(MCQ_m, N_{f_n})$  and update  $\Omega_m^{(x)}$  and  $N_f^{(x)}$
- 11:             **end if**
- 12:     **end for**
- 13:     update  $\Omega_r^{(x)}$
- 14:     update  $CQ^{(x)} = \{\Omega_m^{(x)}, \Omega_r^{(x)}\}$
- 15: **end while**
- 16: in the last loop  $|\Omega_m| < k$  so return final  $CQ = \{\Omega_m^{(x-1)}, \Omega_r^{(x-1)}\}$
- 17: **for all**  $z \in \{1, \dots, |CQ|\}$  **do**
- 18:     **if**  $size(CQ_z) < MinPts$  **then**
- 19:          $break(CQ_z)$  and update  $CQ$  and  $N_f$
- 20:     **end if**
- 21: **end for**
- 22: expand all CQs by the CP expansion rule

---

FIGURE 3.11: PSEUDOCODE OF CQ MERGING ALGORITHM

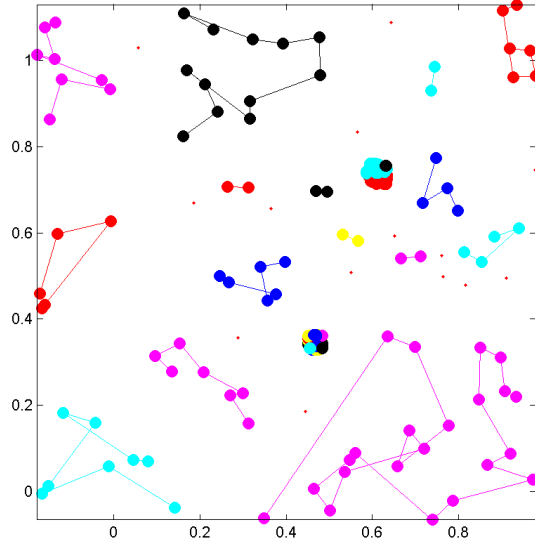


FIGURE 3.12: CQ MERGING RESULT

with a smaller  $\kappa$ . In other words,  $R(CQ_i, CQ_j)$  is a dissimilarity measure as a quotient of the single-linkage by the more decisive  $\kappa$ .

The CQ set is updated by merging two clusters with the minimum  $R$

$$\min_{\{CQ_j \in \mathcal{N}(CQ_i) \vee CQ_i \in \mathcal{N}(CQ_j)\}} \left\{ R(CQ_i, CQ_j) \right\} \quad (3.12)$$

The pseudocode of the strong merging algorithm is shown in Figure 3.13. The clustering is shown in Figure 3.14. There are 4 clusters and 1 noise node. If we set  $MinPts = 6$ , the yellow cluster will be broken and merged to the blue cluster.

### 3.5 Algorithm Description

The M1NN agglomerative clustering algorithm is given as:

1. Given the data set and cluster number  $k$ , find all the CPs.
2. Expand CPs to CQs by the algorithm described in section 3.4.1.
3. Update the CQ set by the CQ merging algorithm in Figure 3.11.
4. Apply the strong merging rule in Figure 3.13 for approximately  $k$  clusters.

---

**Algorithm 2** Strong merging algorithm

---

**Require:** user-defined cluster number  $k$

Euclidean distance function between data points  $d(.,.)$   
compatibility function between clusters  $R(.,.)$   
cluster function to merge clusters or nodes to one cluster  $merge()$   
initial  $CQ^{(0)}$  set and free nodes set  $N_f^{(0)}$   
initialize  $x = 0$

- 1: **while**  $|CQ| > k$  **do**
- 2:   construct  $k$ -NN graph on  $CQ$
- 3:    $x = x + 1$
- 4:   **for all**  $(i, j) \in \{1, \dots, |CQ|\}$  **do**
- 5:     **if**  $R(CQ_i, CQ_j) < R(CQ_i, CQ_a) \wedge R(CQ_i, CQ_j) < R(CQ_j, CQ_b) \wedge$   
     $\{a, b\} \neq \{i, j\} \wedge CQ_j \in \mathcal{N}(CQ_i) \wedge CQ_i \in \mathcal{N}(CQ_j)$  **then**
- 6:        $merge(CQ_i, CQ_j)$  and update  $CQ^{(x)}$
- 7:     **end if**
- 8:   **end for**
- 9:   **for all**  $m \in \{1, \dots, |CQ|\} \wedge n \in \{1, \dots, |N_f|\}$  **do**
- 10:     **if**  $\kappa(CQ_m) > \kappa(\{N_{f_n}, CQ_m\}) \vee \alpha * \kappa(CQ_m) \geq d(N_{f_n}, CQ_m)$  **then**
- 11:        $merge(CQ_m, N_{f_n})$  and update  $CQ^{(x)}$  and  $N_f^{(x)}$
- 12:     **end if**
- 13:   **end for**
- 14: **end while**

---

FIGURE 3.13: PSEUDOCODE OF STRONG MERGING ALGORITHM

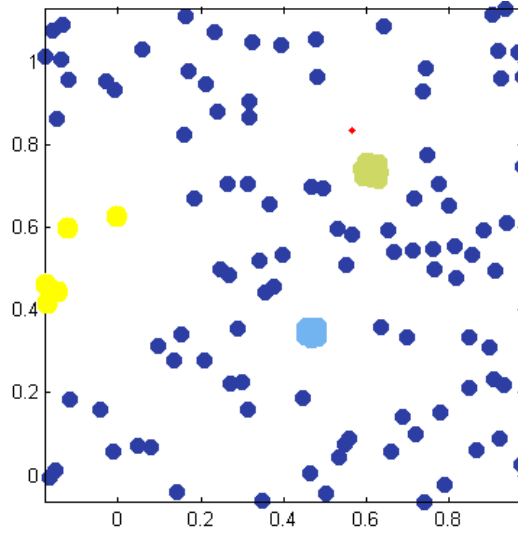


FIGURE 3.14: STRONG MERGING RESULT



## 3.6 Experimental Analysis

In this section, the experimental analysis of the M1NN agglomerative clustering method is presented. The clustering process is demonstrated on two challenging data sets. The clustering performance is investigated for chaining phenomenon and data variations. The clustering methods introduced earlier, i.e. KM (K-means) [68], AP (Affinity Propagation) [45], NCUT (Normalized Cut) [2], ISOPM (Isoperimetric Partitioning) [59], HC (Hierarchical Clustering) [27], CHA (CHAMELEON) [33] and DBSCAN [50], are quantitatively compared with our method on different types of data.

### 3.6.1 Clustering Process Demonstration

In this section, the clustering process is visualized. Figure 3.15 shows the merging steps for a complex data set. As can be seen, there are hundreds of small CQs at the beginning of clustering. The CQs are merged iteratively by our method and 3 spiral clusters become visible gradually. In this demonstration, MCQs and RCQs are denoted by colorful nodes and circles respectively.

Figure 3.16 shows the clustering process on a data set containing two three-dimensional intervened rings and many outliers around. A semi-supervised relaxation labeling method [157] was also applied to this data set but only 60% of all data points were correctly clustered.

### 3.6.2 Chaining Phenomenon

In this section, clustering methods are applied to a data set with the chaining phenomenon problem [30]. The clustering results are shown in Figure 3.17. Our method without the strong merging rule gives a correct but slightly sparse clustering. The strong merging rule improves the clustering for exactly 3 clusters as

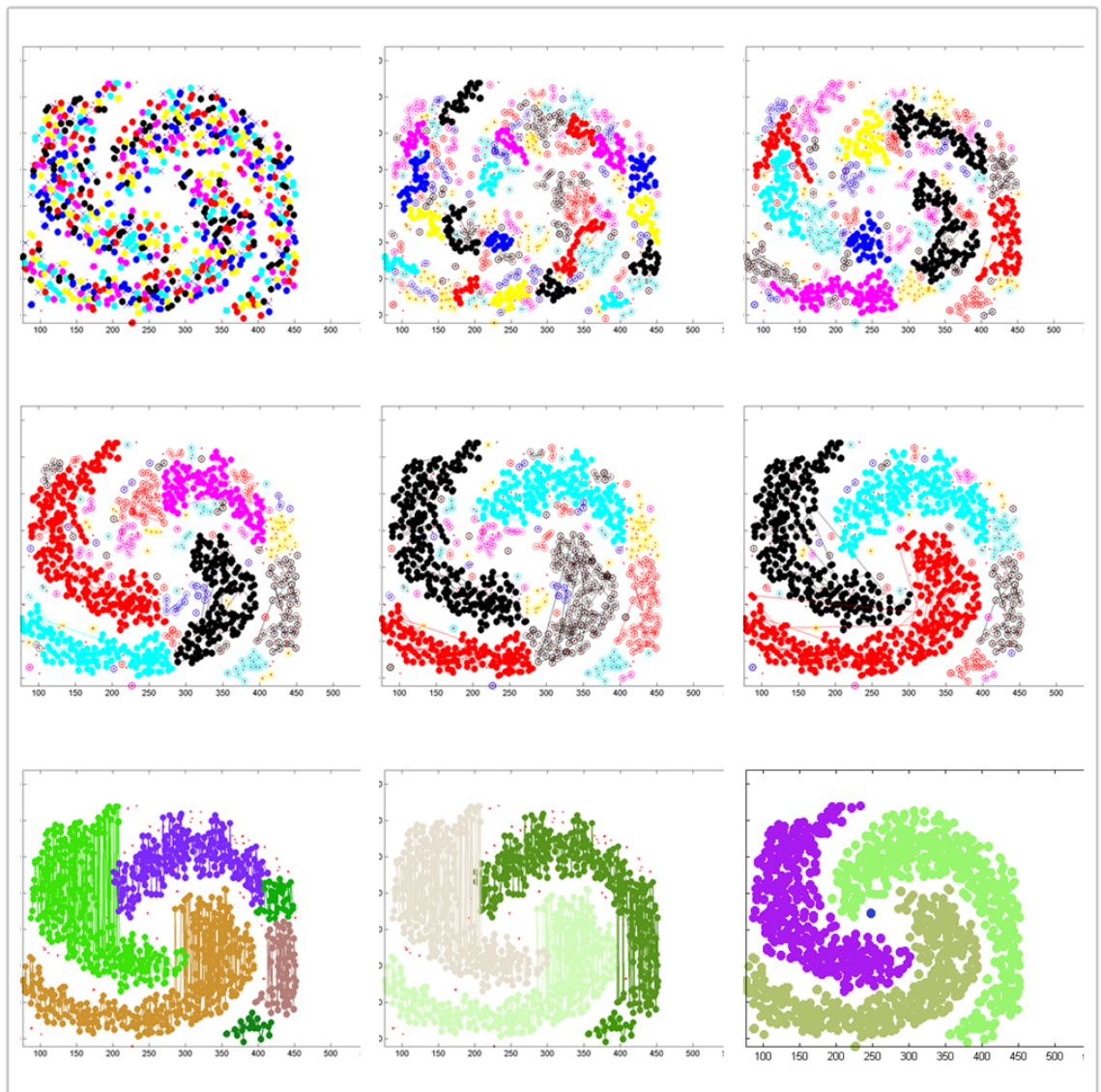


FIGURE 3.15: CLUSTERING DEMONSTRATION 1

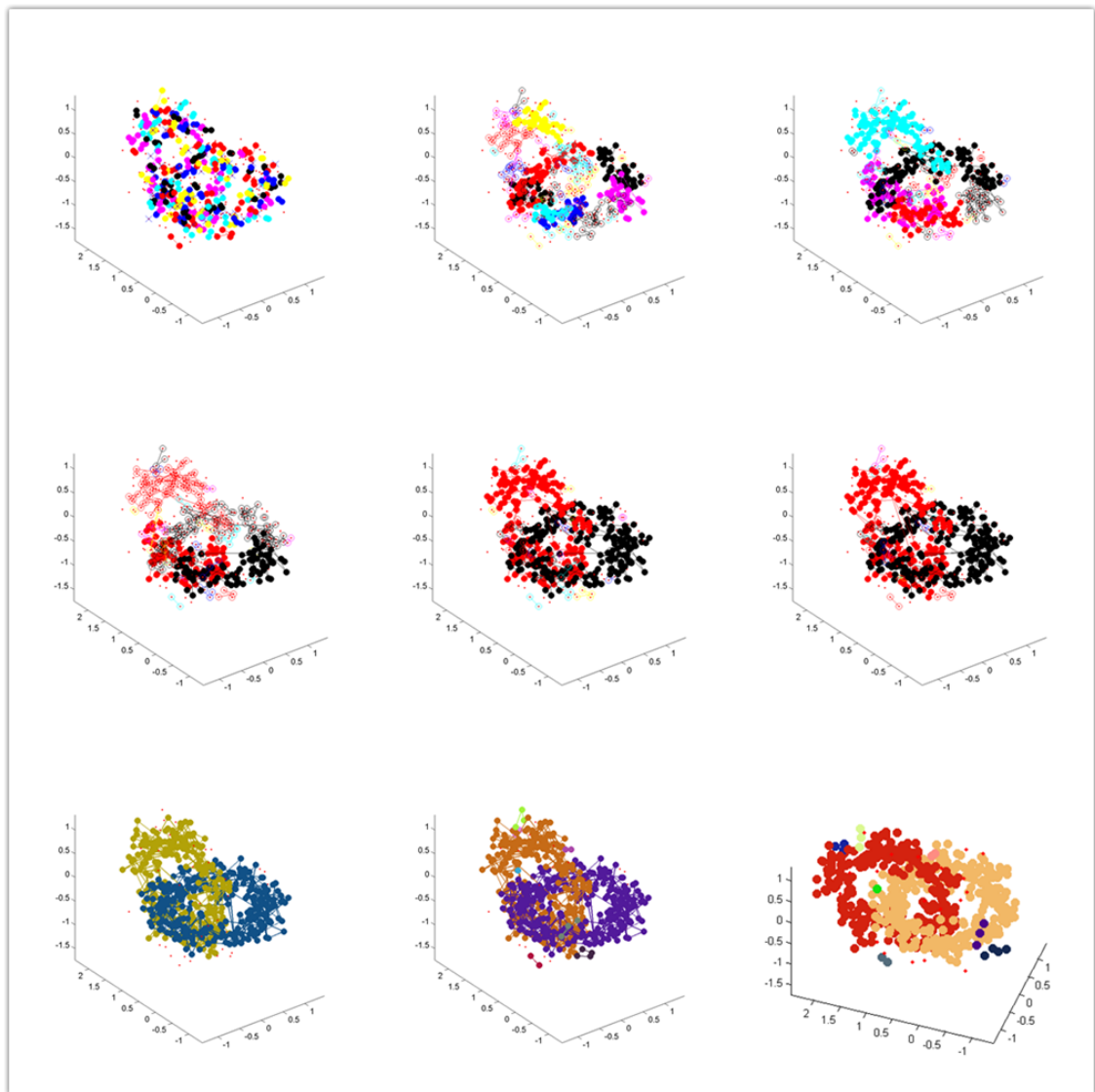


FIGURE 3.16: CLUSTERING DEMONSTRATION 2

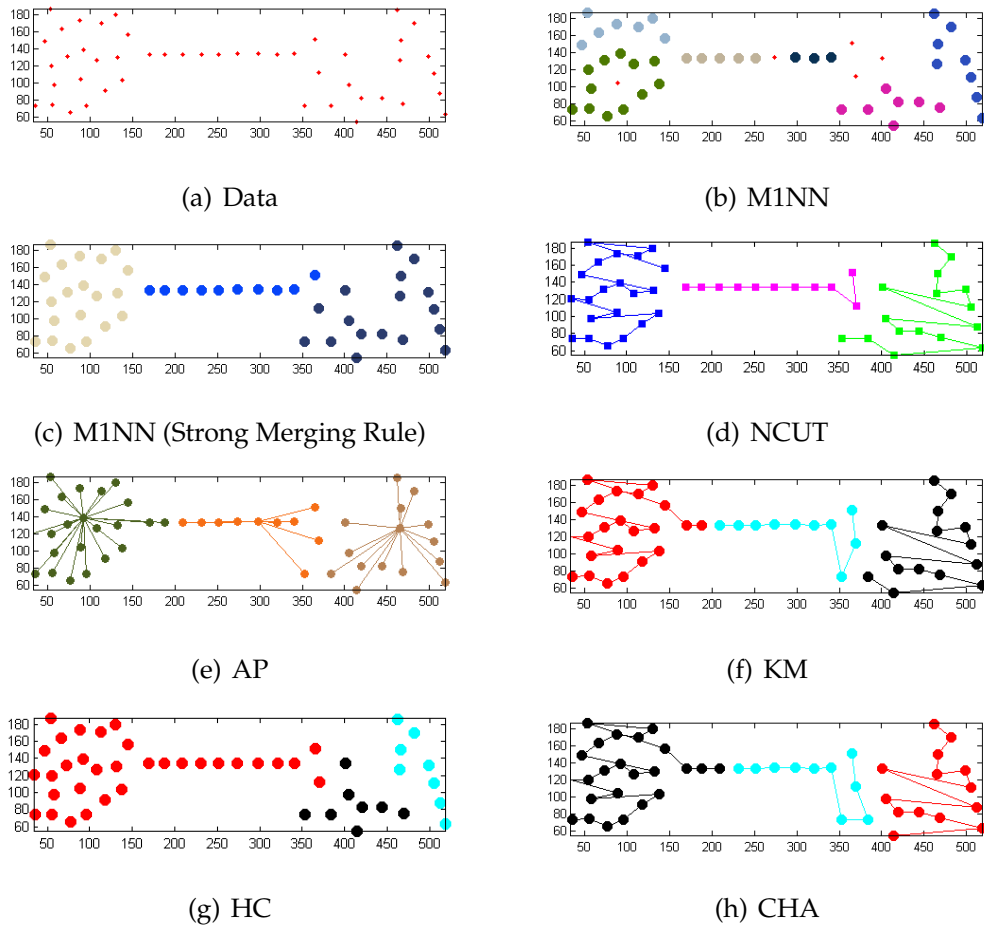


FIGURE 3.17: CHAINING PHENOMENON DEMONSTRATION

shown in Figure 3.17(c). NCUT has the second best performance. KM and AP find the same clusters and are marginally outperformed by NCUT. On the other hand, the classical HC suffers heavily from the chaining phenomenon whilst CHA as an enhanced hierarchical clustering method has a better result, although which is still not good enough in comparison with the other methods.

### 3.6.3 York2012

In this section, a toy data set York2012 is created. It contains simple variations to test the consistency of the clustering methods. The first experiment is shown

in Figure 3.18. While most methods, e.g. ISOPM and CHA, are capable of this easy task, KM and AP generate quite erroneous results. In Figure 3.18(c), 'O' is divided into two parts which are assigned to 'Y' and 'R', and '0' is split as two clusters. The incorrectness is possibly caused by the random initialization of KM. In Figure 3.18(d), some nodes of 'O' and '0' are grabbed by the exemplars in 'Y' and '1', which are identified by AP as centers therefore are strongly attractive to the other nodes in a certain range.

In the second experiment, an elongated cluster is added at the bottom which affects KM, AP, NCUT and CHA badly. Figure 3.19(c) shows the cluster is cut into two pieces by KM, which are merged into '2' and the lower half of '0'. In the meantime, the upper half of '0' is grouped together with '1'. Figure 3.19(d) demonstrates AP is unable to handle elongated clusters thus its performance is worsened by the new setting. NCUT is less influenced although the left end of the additional cluster is taken off and wrongly assigned to '2' as shown in Figure 3.19(e). Surprisingly, as Figure 3.19(h) shows, CHA is only slightly better than AP and is worse than HC. This result may imply the advantage of using a  $k$ -NN graph in a hierarchical clustering method is not always obvious, the failure of which could sometimes cause more problems.

In the third experiment, a hollow rectangular cluster is further added. Due to their nature, KM and AP cannot detect co-centric clusters as shown in Figure 3.20(c) and 3.20(d). Meanwhile, Figure 3.20(e) indicates that NCUT has degenerated massively and can only capture very few clusters, i.e. 'K' and some segments of the rectangle. On the other hand, Figure 3.20(h) shows that CHA outperforms NCUT greatly by having many correct clusters, although the numbers '2', '1' and '2' are still heavily mixed with the elongated cluster.

In the fourth experiment, 50 uniformly distributed nodes are added. Figure 3.21(b) shows that M1NN works satisfactorily. Figure 3.21(c) presents the second

best method DBSCAN and its clustering, in which detected outliers are marked with red circles. Apparently, M1NN outperforms DBSCAN by accurately identifying the outliers close to 'Y', '1' and the last '2'. ISOPM cannot distinguish outliers from cluster members however the clustering is reasonably good, although a little sparse as shown in Figure 3.21(f). Figure 3.21(h) shows CHA is still struggling with the elongated cluster, nevertheless its performance is better than HC by separating the first '2' from '0'. As can be seen in Figure 3.21(g), there are also 3 trivial clusters generated by HC due to its defect illustrated in Figure 3.5(b). Comparing with the unusable AP, NCUT has 'O' and 'K' correctly clustered at its best.

### 3.6.4 Experimental Comparison

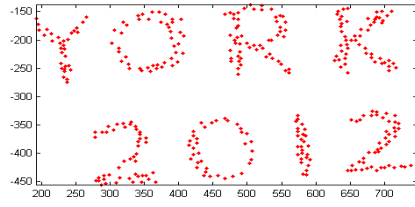
In this section, the quantitative comparison of clustering methods is presented. RI (Rand Index) [158] is used to assess a clustering against the ground truth and is defined as

$$RI = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.13)$$

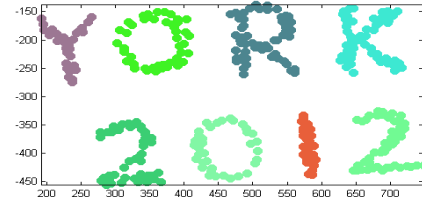
where TP is the number of *true positives*. TN is the number of *true negatives*. FP is the number of *false positives* and FN is the number of *false negatives*. TP and TN together are termed *agreement*. FP and FN together are called *disagreement*. RI = 1 means the clustering is identical to the ground truth.

Since using one index may not be enough [159], MER (Mean Error Rate) is proposed to quantitatively measure the errors in a clustering. Given a cluster of size  $m$ , in which  $n$  data points are mis-assigned, the error rate is  $n/m$  for the cluster. For  $k$  clusters, MER is defined as

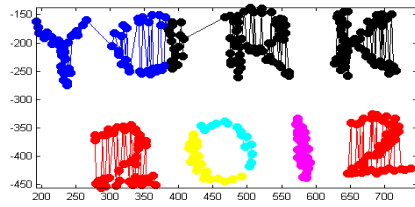
$$MER = \frac{1}{k} \sum_{i=1}^k \frac{n_i}{m_i} \quad (3.14)$$



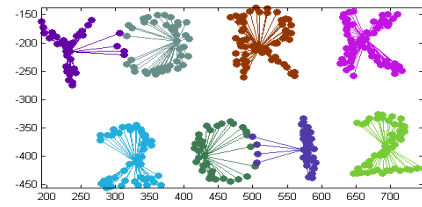
(a) Data



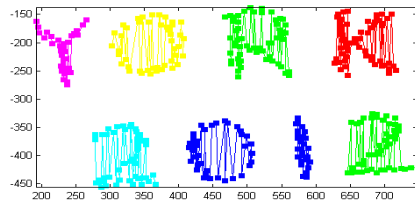
(b) M1NN



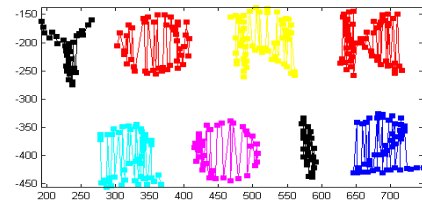
(c) KM



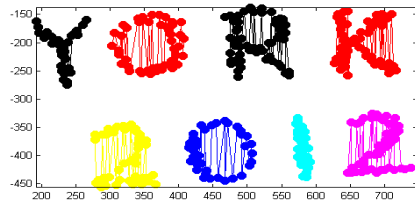
(d) AP



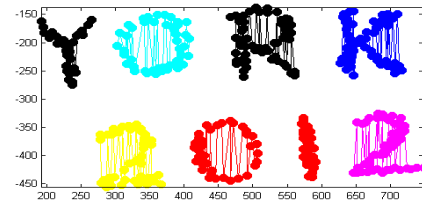
(e) NCUT



(f) ISOPM

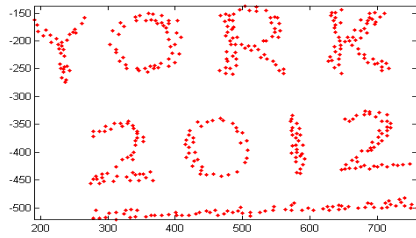


(g) HC

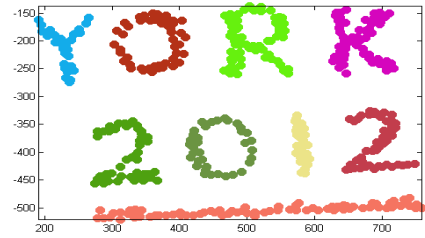


(h) CHA

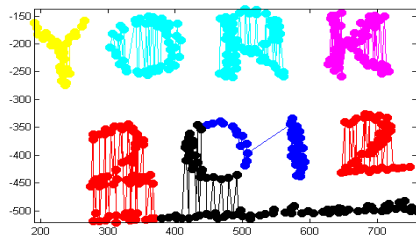
FIGURE 3.18: YORK2012 CLUSTERING COMPARISON 1



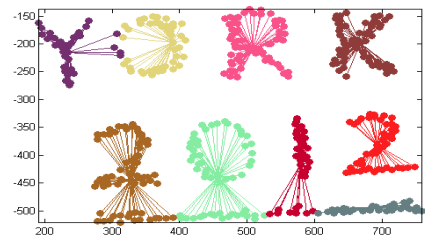
(a) Data



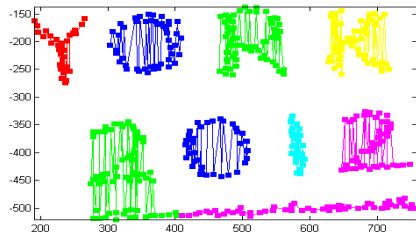
(b) M1NN



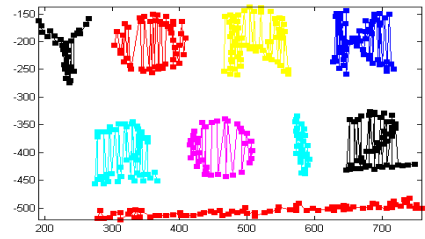
(c) KM



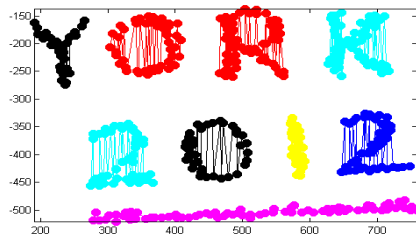
(d) AP



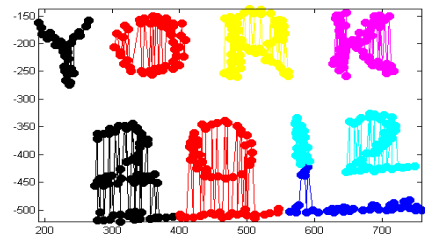
(e) NCUT



(f) ISOPM



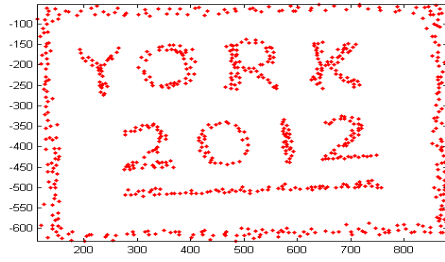
(g) HC



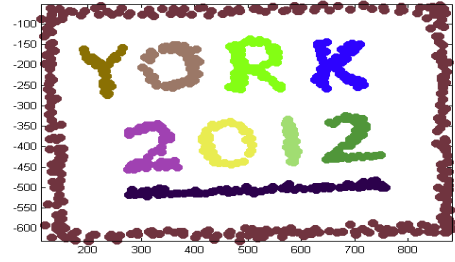
(h) CHA

FIGURE 3.19: YORK2012 CLUSTERING COMPARISON 2

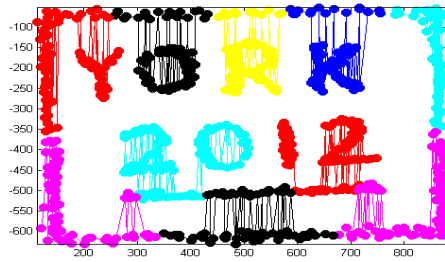




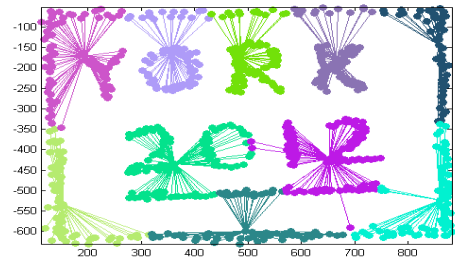
(a) Data



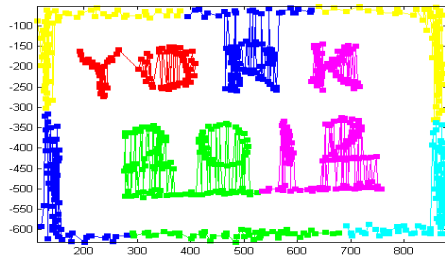
(b) M1NN



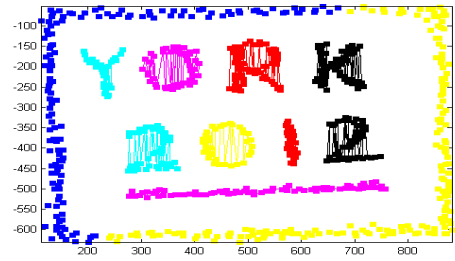
(c) KM



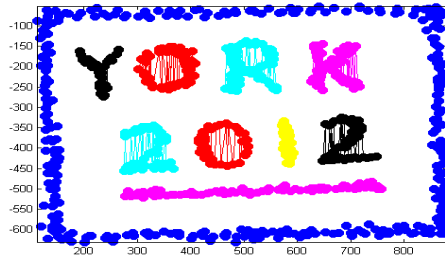
(d) AP



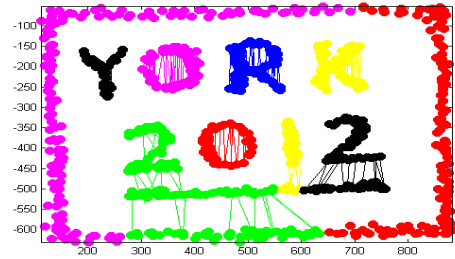
(e) NCUT



(f) ISOPM

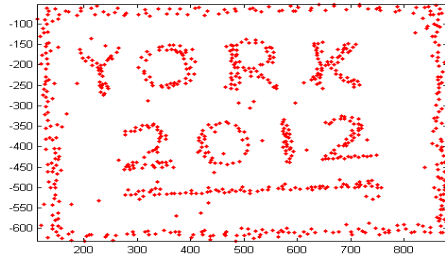


(g) HC

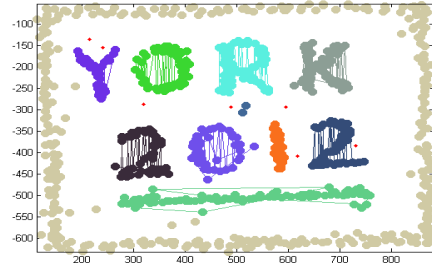


(h) CHA

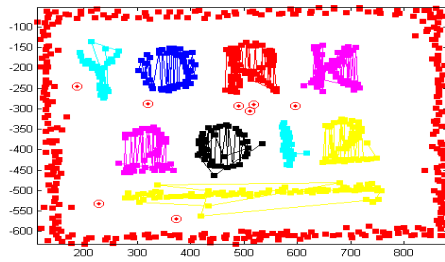
FIGURE 3.20: YORK2012 CLUSTERING COMPARISON 3



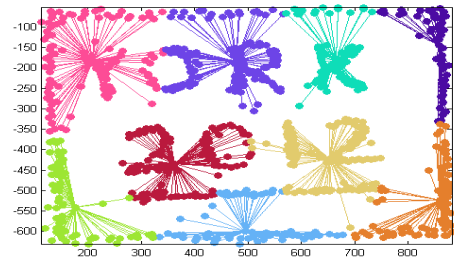
(a) Data



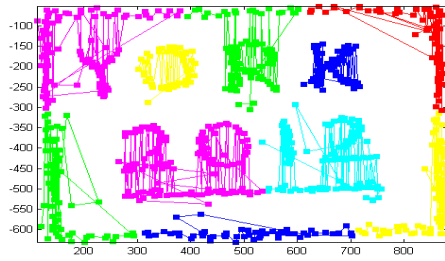
(b) M1NN



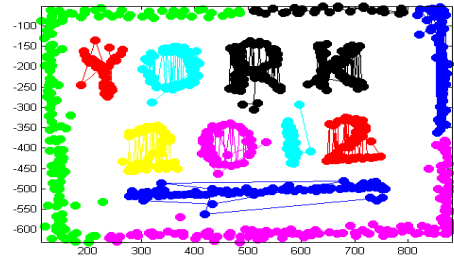
(c) DBSCAN ( $k=6$ )



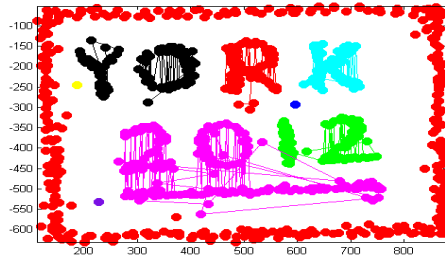
(d) AP



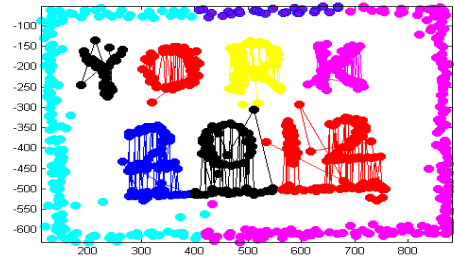
(e) NCUT



(f) ISOPM



(g) HC



(h) CHA

FIGURE 3.21: NOISY YORK2012 CLUSTERING COMPARISON

In the first experiment, test data are point patterns in the Euclidean space, which are listed in Table 3.1. The data sets NC1, NC2 and NC3 are used for data clustering demonstration by NCUT [2]. The data set Complex8 is built to test clustering of 2-D patterns with variable structure [160]. The other data sets [161] are used for spectral [162, 163] and kernel K-means clustering or relaxed labeling [157]. The RI and MER values are drawn in Table 3.2 and 3.3. As can be seen, our method has the best performance overall. In Table 3.2, the RI scores of M1NN are close to 1 for most data sets. NCUT, ISOPM and CHA are in the second tier. NCUT beats the other two methods in NC2, NC3, Caltech3 and Caltech5. ISOPM performs better in Caltech2, Caltech4, Complex8 and RG. In the meantime, CHA outdoes NCUT and ISOPM in Caltech1, Caltech6 and 2R. HC has some best marks but its performance drops radically due to the chaining phenomenon problem, which is found in Caltech2 and Caltech4. KM and AP have similar clusterings and are ranked the least useful clustering methods by their RI scores. In Table 3.3, the MER scores of M1NN are lowest in general. ISOPM, as the second best method, is followed by CHA, HC and NCUT which have low and acceptable error rates on average and are in the same group. The last two methods are still KM and AP.

In the second experiment, non-Euclidean data sets in the form of long feature vectors and pairwise distance matrices are tested, e.g. Shape25 in Figure 3.22 which is used for learning shape-classes [164]. Because the strong merging rule does not work properly in a non-Euclidean space, the input cluster number for the other methods is set equal to the output of our method, which is bigger than the user-defined cluster number  $k$ . KM is also replaced by KMD (K-medoids). Table 3.4 shows that our method achieves the highest RI scores overall. KMD, AP, CHA and NCUT are in the second class with similar results. HC outperforms ISOPM marginally but both are reckoned inappropriate for non-Euclidean data.

TABLE 3.1: THE CHARACTERISTICS OF DATA SETS















| Data     | Icon  | Number of clusters | Number of objects in each cluster |
|----------|---|--------------------|-----------------------------------|
| NC1      |    | 2                  | [40,50]                           |
| NC2      |    | 3                  | [80,10,10]                        |
| NC3      |    | 4                  | [80,10,20,20]                     |
| Caltech1 |    | 3                  | [61,139,99]                       |
| Caltech2 |    | 3                  | [106,102,95]                      |
| Caltech3 |   | 3                  | [118,75,73]                       |
| Caltech4 |  | 5                  | [136,116,111,150,109]             |
| Caltech5 |  | 4                  | [117,123,150,122]                 |
| Caltech6 |  | 3                  | [56,82,100]                       |
| Complex8 |  | 8                  | [10,200,200,100,100,50,50,29]     |
| 2S       |  | 2                  | [193,193]                         |
| RD       |  | 2                  | [500,100]                         |
| RG       |  | 3                  | [100,150,40]                      |
| 2R       |  | 2                  | [305,295]                         |

TABLE 3.2: RI OF CLUSTERING METHODS

|          | M1NN  | KM   | AP   | NCUT  | ISOPM | HC   | CHA   |
|----------|-------|------|------|-------|-------|------|-------|
| NC1      | 0.95  | 1    | 1    | 1     | 0.87  | 1    | 1     |
| NC2      | 1     | 0.47 | 0.45 | 0.78  | 0.46  | 1    | 0.49  |
| NC3      | 1     | 0.67 | 0.66 | 1     | 0.70  | 1    | 0.92  |
| Caltech1 | 1     | 0.56 | 0.58 | 0.73  | 0.69  | 1    | 0.85  |
| Caltech2 | 0.99  | 0.75 | 0.74 | 0.74  | 0.85  | 0.36 | 0.83  |
| Caltech3 | 1     | 0.73 | 0.72 | 1     | 0.83  | 1    | 0.93  |
| Caltech4 | 0.99  | 0.88 | 0.89 | 0.89  | 0.91  | 0.23 | 0.86  |
| Caltech5 | 1     | 0.80 | 0.80 | 1     | 0.97  | 1    | 0.92  |
| Caltech6 | 0.996 | 0.83 | 0.83 | 0.90  | 0.86  | 0.80 | 0.98  |
| Complex8 | 0.98  | 0.80 | 0.81 | 0.81  | 0.996 | 0.79 | 0.91  |
| 2S       | 1     | 0.50 | 0.50 | 0.50  | 0.54  | 1    | 0.50  |
| RD       | 0.997 | 0.50 | 0.50 | 0.997 | 0.33  | 0.72 | 0.997 |
| RG       | 1     | 0.62 | 0.62 | 0.55  | 0.99  | 0.41 | 0.64  |
| 2R       | 0.95  | 0.56 | 0.56 | 0.60  | 0.82  | 0.50 | 0.98  |

TABLE 3.3: MER OF CLUSTERING METHODS

|          | M1NN  | KM   | AP   | NCUT  | ISOPM | HC   | CHA   |
|----------|-------|------|------|-------|-------|------|-------|
| NC1      | 0     | 0    | 0    | 0     | 0     | 0    | 0     |
| NC2      | 0     | 0.18 | 0.19 | 0.14  | 0     | 0    | 0.10  |
| NC3      | 0     | 0.20 | 0.21 | 0     | 0     | 0    | 0.20  |
| Caltech1 | 0     | 0.49 | 0.49 | 0.19  | 0.06  | 0    | 0.09  |
| Caltech2 | 0     | 0.19 | 0.19 | 0.20  | 0.02  | 0.22 | 0.10  |
| Caltech3 | 0     | 0.21 | 0.22 | 0     | 0     | 0    | 0     |
| Caltech4 | 0     | 0.16 | 0.15 | 0.14  | 0.04  | 0.15 | 0.09  |
| Caltech5 | 0     | 0.18 | 0.19 | 0     | 0     | 0    | 0     |
| Caltech6 | 0     | 0.12 | 0.13 | 0.07  | 0.07  | 0.15 | 0.01  |
| Complex8 | 0     | 0.44 | 0.30 | 0.31  | 0.02  | 0.20 | 0.02  |
| 2S       | 0     | 0.50 | 0.50 | 0.50  | 0.20  | 0    | 0.50  |
| RD       | 0.001 | 0.16 | 0.16 | 0.001 | 0.007 | 0.08 | 0.001 |
| RG       | 0     | 0.27 | 0.27 | 0.47  | 0.004 | 0.16 | 0.12  |
| 2R       | 0.02  | 0.33 | 0.32 | 0.24  | 0.03  | 0.24 | 0.01  |

TABLE 3.4: RI OF CLUSTERING METHODS

|                | M1NN  | KMD   | AP    | NCUT  | ISOPM | HC   | CHA   |
|----------------|-------|-------|-------|-------|-------|------|-------|
| Iris           | 0.83  | 0.74  | 0.74  | 0.71  | 0.77  | 0.77 | 0.76  |
| Wine           | 0.80  | 0.72  | 0.72  | 0.71  | 0.34  | 0.40 | 0.74  |
| Shape25        | 0.90  | 0.83  | 0.84  | 0.85  | 0.08  | 0.63 | 0.97  |
| ChickenPieces1 | 0.83  | 0.79  | 0.78  | 0.81  | 0.21  | 0.33 | 0.75  |
| ChickenPieces2 | 0.82  | 0.79  | 0.80  | 0.81  | 0.21  | 0.39 | 0.80  |
| ChickenPieces3 | 0.84  | 0.81  | 0.81  | 0.81  | 0.21  | 0.37 | 0.78  |
| ChickenPieces4 | 0.86  | 0.82  | 0.81  | 0.81  | 0.21  | 0.38 | 0.82  |
| PenDigits1     | 0.913 | 0.909 | 0.909 | 0.907 | 0.10  | 0.38 | 0.909 |
| PenDigits2     | 0.914 | 0.911 | 0.913 | 0.908 | 0.10  | 0.43 | 0.909 |
| PenDigits3     | 0.914 | 0.910 | 0.910 | 0.905 | 0.10  | 0.45 | 0.909 |
| PenDigits4     | 0.908 | 0.903 | 0.903 | 0.903 | 0.10  | 0.38 | 0.909 |

TABLE 3.5: MER OF CLUSTERING METHODS

|                | M1NN  | KMD  | AP   | NCUT | ISOPM | HC   | CHA  |
|----------------|-------|------|------|------|-------|------|------|
| Iris           | 0.02  | 0.05 | 0.04 | 0.14 | 0.26  | 0.03 | 0.08 |
| Wine           | 0.02  | 0.06 | 0.08 | 0.12 | 0.60  | 0.03 | 0.03 |
| Shape25        | 0.10  | 0.38 | 0.39 | 0.41 | 0.88  | 0.27 | 0.06 |
| ChickenPieces1 | 0.04  | 0.25 | 0.14 | 0.31 | 0.74  | 0.02 | 0.21 |
| ChickenPieces2 | 0.02  | 0.12 | 0.10 | 0.23 | 0.74  | 0.01 | 0.05 |
| ChickenPieces3 | 0.01  | 0.10 | 0.05 | 0.19 | 0.74  | 0.02 | 0.05 |
| ChickenPieces4 | 0.002 | 0.11 | 0.06 | 0.20 | 0.74  | 0.02 | 0.04 |
| PenDigits1     | 0.04  | 0.10 | 0.07 | 0.18 | 0.90  | 0.01 | 0.09 |
| PenDigits2     | 0.01  | 0.04 | 0.02 | 0.12 | 0.90  | 0.01 | 0.08 |
| PenDigits3     | 0.02  | 0.07 | 0.05 | 0.16 | 0.90  | 0.01 | 0.08 |
| PenDigits4     | 0.12  | 0.24 | 0.17 | 0.26 | 0.90  | 0.02 | 0.08 |



On the other hand, Table 3.5 shows the MER scores of M1NN are the lowest generally. HC, AP, KMD and CHA also have lower error rates comparing with NCUT. ISOPM gives the worst performance in this experiment.
































The clusterings of Iris and Shape25 from M1NN are further inspected. For Iris, our method has the same problem as Mercer kernel based clustering method [146]. The error could be caused by the non-numeric distance definition rather than clustering algorithms. For Shape25, the clustering by our method is compared with the original work [165] in Figure 3.23. The incorrect cluster members are marked in squares.

### 3.6.5 Noise Test

In this section, a noise test is conducted on Caltech1 to compare the clustering methods M1NN and DBSCAN. Each time 10 more uniformly distributed nodes are added to the data set. Since there are no labels with noise nodes and RI and MER cannot be used, two new measures are proposed. The first counts the number of errors when a cluster is partially merged into another. If such a merging is avoidable by breaking a cluster ('under' clustering), the number of total clusters is shown in the parenthesis. The second counts the number of noise nodes detected. The result is shown in Table 3.6. The clusterings of Caltech1 with 150 noise nodes from both methods are demonstrated in Figure 3.24. Obviously, our method outperforms DBSCAN on detecting the central cluster whilst keeping the outer ring cluster against heavy noise.

## 3.7 Advanced Applications

In this section, our method is adapted for advanced applications, e.g. data mining, image segmentation and manifold learning. For each application, our method

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|
|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|

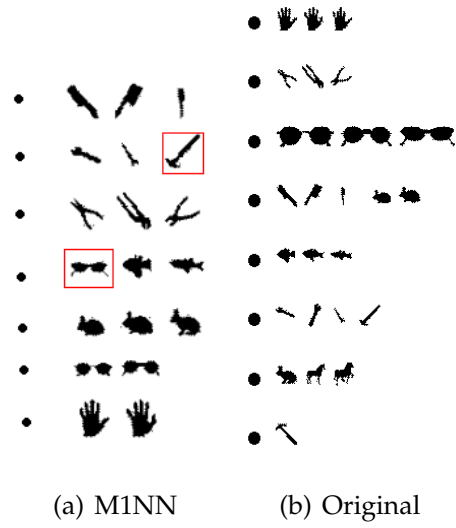


FIGURE 3.23: CLUSTERING COMPARISON FOR SHAPE25

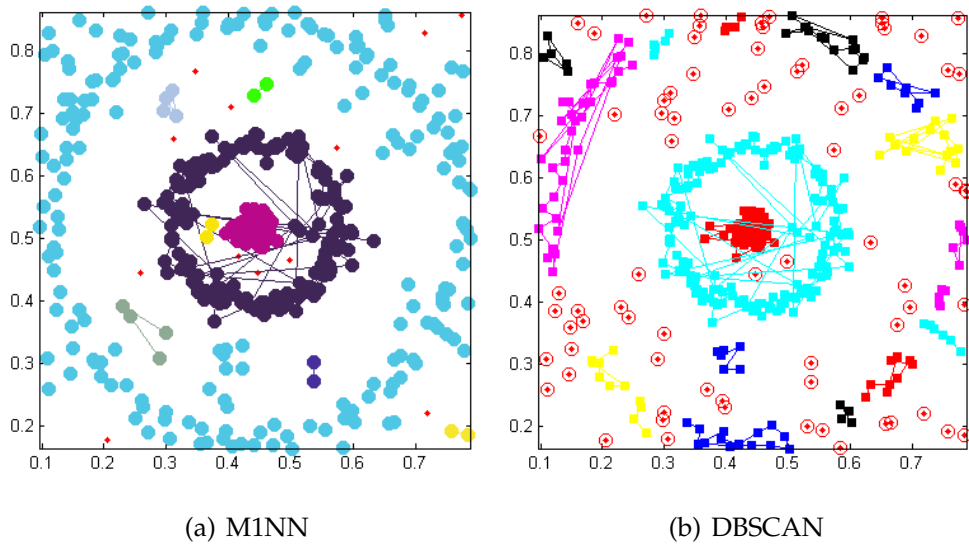


FIGURE 3.24: CLUSTERINGS OF M1NN AND DBSCAN UNDER HEAVY NOISE

TABLE 3.6: NOISE TEST BETWEEN M1NN AND DBSCAN

| Caltech1 (299 nodes) | M1NN  |       | DBSCAN |       |
|----------------------|-------|-------|--------|-------|
|                      | Error | Noise | Error  | Noise |
| 0 nodes added        | 0     | 6     | 0      | 0     |
| 10 nodes added       | 0     | 7     | 0      | 1     |
| 20 nodes added       | 0     | 4     | 0      | 2     |
| 30 nodes added       | 0     | 5     | 0      | 3     |
| 40 nodes added       | 0     | 6     | 0      | 6     |
| 50 nodes added       | 0     | 12    | 0(7)   | 25    |
| 60 nodes added       | 0     | 11    | 0(7)   | 22    |
| 70 nodes added       | 0     | 11    | 0(16)  | 48    |
| 80 nodes added       | 0     | 9     | 0(17)  | 55    |
| 90 nodes added       | 0     | 7     | 0(16)  | 59    |
| 100 nodes added      | 0     | 11    | 0(16)  | 63    |
| 110 nodes added      | 0     | 11    | 0(17)  | 68    |
| 120 nodes added      | 0     | 6     | 0(17)  | 66    |
| 130 nodes added      | 0     | 9     | 0(17)  | 76    |
| 140 nodes added      | 0     | 10    | 0(16)  | 82    |
| 150 nodes added      | 0     | 12    | 0(18)  | 74    |

outperforms the standard method and demonstrates promising potential.

### 3.7.1 Data Mining

A flight routing problem was studied by Frey and Dueck [45] using their AP clustering method. They identified a number of Canadian and American cities that are most easily accessible by large subsets of other cities in terms of estimated commercial airline travel time. These pivotal cities are Seattle, Calgary, Minneapolis, Denver, Toronto, Philadelphia and Atlanta, which are the exemplars of 7 clusters. The pivotal cities and clusters are shown in Figure 3.25.

On the other hand, a number of clusters are detected as the most vibrant areas for air travel by M1NN. The first 10 clusters are shown in Figure 3.26. The CA cities, which are split and assigned to different clusters centered at Denver and Seattle by AP, are correctly grouped in one cluster by M1NN. The AK and FL clusters are also accurately recognized by M1NN, in contrast they are mixed with other cities in bigger clusters by AP. Taking a single city for inspection, for example Chicago, it is assigned to the pivotal city Minneapolis by M1NN and to Philadelphia by AP. Geographically, Chicago is much closer to Minneapolis and they are directly connected by several airlines. Hence our method provides more useful and correct information.

### 3.7.2 Image Segmentation

Literally, any clustering method can be adapted for image segmentation tasks. In practice however, only a limited number of clustering methods have achieved good performance. Figure 3.27 shows the comparison of our method with NCUT and ISOPM in segmenting a variety of images. As shown in Figure 3.27(i), 3.27(q) and 3.27(u), M1NN beats NCUT and ISOPM by having more details segmented

|  |                               |                       |                          |                           |                       |                     |                   |                        |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
|--|-------------------------------|-----------------------|--------------------------|---------------------------|-----------------------|---------------------|-------------------|------------------------|-------------------------|-------------------------------|--------------------------|-----------------|-----------------|------------------|------------------|---------------------|------------------|------------------|----------------------|------------------|--------------|---------------|---------------|-----------------------|---------------|-----------------------|-----------------|------------------|------------------------------|----------------|--------------|-----------------|-----------------|-------------------|--------------|---------------|-------------|---------------|------------------|--------------|-------------|---------------------|--|--------------|---------------|-------------------------|--------------|------------------------|------------------|----------------|-----------------|-----------------|-------------|--------------|---------------|-------------------------|--------------------|--------------------|-------------|--------------------------|----------|-------------|------------|-----------|--------------|-------------|----------------|--------------|------------------|-----------|
| Akron/Canton-OH                        | Albany-GA                     | Alexandria-LA         | Asheville-NC             | Oklaoma City-OK           | Ontario-CA            | Orange County-CA    | Oxnard/Ventura-CA |                        |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| <u>Atlanta-GA</u>                      | Atlantic City-NJ              | Augusta-GA            | Austin-TX                | Palm Springs-CA           | Phoenix-AZ            | Pierre-SD           | Ponca City-OK     | Prescott-AZ            |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Baton Rouge-LA                         | Beaumont/Pt. Arthur-TX        | Binghamton-NY         | Birmingham-AL            | Pueblo-CO                 | Rapid City-SD         | Riverton-WY         | Rock Springs-WY   | Salt Lake City-UT      |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Bloomington-IL                         | Brownsville-TX                | Brunswick-GA          | Burlington-VT            | San Angelo-TX             | San Antonio-TX        | San Diego-CA        | Santa Barbara-CA  | Santa Fe-NM            |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Cape Girardeau-MO                      | Charleston-SC                 | Charlotte-NC          | Charlottesville-VA       | Santa Maria-CA            | Scottsbluff-NE        | Sheridan-WY         | Show Low-AZ       | St. George-UT          |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Chattanooga-TN                         | College Station-TX            | Columbia-SC           | Columbus-GA              | Steamboat Springs-CO      | Telluride-CO          | Texarkana-AR        | Tucson-AZ         | Tulsa-OK               |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Columbus/Starkville/West Point-MS      | Corpus Christi-TX             | Dayton-OH             | Dothan-AL                | Twin Falls-ID             | Tyler-TX              | Vail-CO             | Wichita-KS        | Williston-ND           | Worland-WY              | Yuma-AZ                       |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Daytona Beach-FL                       | Decatur-IL                    | Del Rio-TX            | Fayetteville-NC          | Aberdeen-SD               | Appleton-WI           | Bemidji-MN          | Bismarck-ND       | Brainerd-MN            |                         |                               |                          |                 |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Escanaba-MI                            | Evansville-IN                 | Fayetteville-AR       | Fayetteville-NC          | Cedar Rapids/Iowa City-IA | Champaign-IL          | Columbia-MO         | Des Moines-IA     | Devils Lake-ND         | Duluth/Superior-MN      | Eau Claire-WI                 | Elmira/Corning-NY        | Fargo-ND        |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Florence-SC                            | Ft. Lauderdale-FL             | Ft. Leonard Wood-MO   | Ft. Myers-FL             | Flint-MI                  | Ft. Dodge-IA          | Grand Forks-ND      | Grand Rapids-MI   | Great Bend-KS          | Green Bay-WI            | Hancock-MI                    | Hibbing/Chisholm-MN      | Idaho Falls-ID  |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Smith-AR                               | Ft. Walton Beach-FL           | Ft. Wayne-IN          | Gainesville-FL           | Green Bay-WI              | Hancock-MI            | Hibbing/Chisholm-MN | Idaho Falls-ID    | International Falls-MN | Jamestown-ND            | Joplin-MO                     | Kalamazoo-MI             | Kansas City-MO  |                 |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Greensboro/High Point/Winston Salem-NC | Greenville-MS                 | Greenville-NC         | Harlingen-TX             | Kingston-ON               | Kitchener/Waterloo-ON | La Crosse-WI        | Lansing-MI        | Laurel/Hattiesburg-MS  | Lincoln-NE              | Madison-WI                    | Marion-IL                | Marquette-MI    | Mason City-IA   |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Greenville/Spartanburg-SC              | Huntington-WV                 | Huntsville/Decatur-AL | Ithaca-NY                | Lincoln-NE                | Madison-WI            | Marion-IL           | Marquette-MI      | Mason City-IA          | Milwaukee-WI            | <u>Minneapolis/St Paul-MN</u> | Minot-ND                 | Moline-IL       | Moses Lake-WA   |                  |                  |                     |                  |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Houston-TX                             | Jackson-MS                    | Jacksonville-FL       | Jacksonville-NC          | Kirksville-MO             | Knoxville-TN          | Lafayette-LA        | Lake Charles-LA   | Latrobe-PA             | Lexington-KY            | Little Rock-AR                | Long Island MacArthur-NY | Longview-TX     | Lubbock-TX      | Lynchburg-VA     | Macon-GA         | Manchester-NH       | McAllen-TX       |                  |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Melbourne-FL                           | Memphis-TN                    | Meridian-MS           | Miami-FL                 | Midland/Odessa-TX         | Mobile-AL             | Monroe-LA           | Montgomery-AL     | Myrtle Beach-SC        | Naples-FL               | Nashville-TN                  | New Bern-NC              | New Orleans-LA  | Newport News-VA | Orlando-FL       | Paducah-KY       | Panama City-FL      | Pensacola-FL     | Peoria-IL        | Portland-ME          |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Raleigh/Durham-NC                      | Roanoke-VA                    | Sarasota/Bradenton-FL | Savannah-GA              | Shreveport-LA             | Springfield-IL        | Springfield-MO      | St. Louis-MO      | Tallahassee-FL         | Tampa/St. Petersburg-FL | Toledo-OH                     | Tri-City Airport-TX      | Tupelo-MS       | Valdosta-GA     | Victoria-TX      | Waco-TX          | West Palm Beach-FL  | Wichita Falls-TX | Wilmington-NC    |                      |                  |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Abbotsford-BC                          | <u>Calgary-AB</u>             | Castlegar-BC          | Comox-BC                 | Cranbrook-BC              | Edmonton-AB           | Ft. McMurray-AB     | Ft. St. John-BC   | Grande Prairie-AB      | Kamloops-BC             | Kelowna-BC                    | Lethbridge-AB            | Medicine Hat-AB | Penticton-BC    | Prince Rupert-BC | Rankin Inlet-NU  | Regina-SK           | Saskatoon-SK     | Terrace-BC       | Whitehorse-YT        | Yellowknife-NT   |              |               |               |                       |               |                       |                 |                  |                              |                |              |                 |                 |                   |              |               |             |               |                  |              |             |                     |  |              |               |                         |              |                        |                  |                |                 |                 |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Abilene-TX                             | Alamosa-CO                    | Albuquerque-NM        | Alliance-NE              | Amarillo-TX               | Aspen-CO              | Bakersfield-CA      | Billings-MT       | Bozeman-MT             | Burbank-CA              | Casper-WY                     | Cedar City-UT            | Chadron-NE      | Cheyenne-WY     | Clovis-NM        | Cody-WY          | Colorado Springs-CO | Cortez-CO        | Crescent City-CA | Dallas-Fort Worth-TX | <u>Denver-CO</u> | Dickinson-ND | Dodge City-KS | Durango-CO    | El Centro/Imperial-CA | El Paso-TX    | Elko-NV               | Enid-OK         | Farmington-NM    | Ft. Huachuca/Sierra Vista-AZ | Garden City-KS | Gillette-WY  | Grand Canyon-AZ | Grand Island-NE | Grand Junction-CO | Gunnison-CO  | Hays-KS       | Inyokern-CA | Jackson-TN    | Jackson-WY       | Kearney-NE   | Killeen-TX  | Lake Havasu City-AZ | Laramie-WY                             | Laredo-TX    | Las Vegas-NV  | Lawton-OK               | Liberal-KS   | Los Angeles-CA         | McCook-NE        | Monterey-CA    | Montrose-CO     | North Platte-NE |             |              |               |                         |                    |                    |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Albany-NY                              | Allentown/Bethlehem/Easton-PA | Alpena-MI             | Altoona-PA               | Athens-GA                 | Augusta-ME            | Baltimore-MD        | Bangor-ME         | Bar Harbor-ME          | Bradford-PA             | Buffalo-NY                    | Burlington-IA            | Charleston-WV   | Chicago-IL      | Cincinnati-OH    | Clarksburg-WV    | Cleveland-OH        | Columbus-OH      | Detroit-MI       | Dubois-PA            | Dubuque-IA       | Erie-PA      | Franklin-PA   | Greenbrier-WV | Hagerstown-MD         | Harrisburg-PA | Hilton Head Island-SC | Indianapolis-IN | Iron Mountain-MI | Ironwood-MI                  | Jamestown-NY   | Johnstown-PA | Lancaster-PA    | Lebanon-NH      | Louisville-KY     | Manhattan-KS | Manistee-MI   | Massena-NY  | Morgantown-WV | Muscle Shoals-AL | New Haven-CT | New York-NY | Newburgh-NY         | Norfolk/Virginia Beach/Williamsburg-VA | North Bay-ON | Ogdensburg-NY | Parkersburg/Marietta-WV | Pellston-MI  | <u>Philadelphia-PA</u> | Pittsburgh-PA    | Plattsburgh-NY | Presque Isle-ME | Providence-RI   | Richmond-VA | Rochester-NY | Salina-KS     | Salisbury-Ocean City-MD | Sault Ste Marie-MI | Syracuse-NY        |             |                          |          |             |            |           |              |             |                |              |                  |           |
| Washington-DC                          | Watertown-NY                  | Westchester County-NY | Wilkes-Barre/Scranton-PA | Anchorage-AK              | Barrow-AK             | Bellingham-WA       | Bethel-AK         | Boise-ID               | Butte-MT                | Campbell River-BC             | Chico-CA                 | Cordova-AK      | Dillingham-AK   | Eugene-OR        | Eureka/Arcata-CA | Fairbanks-AK        | Fresno-CA        | Great Falls-MT   | Helena-MT            | Hilo-HI          | Honolulu-HI  | Hoolehua-HI   | Juneau-AK     | Kahului-HI            | Kalispell-MT  | Kapalua-HI            | Kenai-AK        | Ketchikan-AK     | Klamath Falls-OR             | Kodiak-AK      | Kona-HI      | Kotzebue-AK     | Lanai City-HI   | Lewiston-ID       | Lihue-HI     | Long Beach-CA | Medford-OR  | Missoula-MT   | Modesto-CA       | Nanaimo-BC   | Nome-AK     | North Bend-OR       | Oakland-CA                             | Pasco-WA     | Pendleton-OR  | Petersburg-AK           | Pocatello-ID | Portland-OR            | Prince George-BC | Pullman-WA     | Quesnel-BC      | Redding-CA      | Redmond-OR  | Reno-NV      | Sacramento-CA | San Francisco-CA        | San Jose-CA        | San Luis Obispo-CA | Sandspr-B-C | <u>Seattle/Tacoma-WA</u> | Sitka-AK | Smithers-BC | Spokane-WA | Valdez-AK | Vancouver-BC | Victoria-BC | Walla Walla-WA | Wenatchee-WA | Williams Lake-BC | Yakima-WA |

(a)

(b)

|                  |                          |                       |                          |                    |
|------------------|--------------------------|-----------------------|--------------------------|--------------------|
| Washington-DC    | Watertown-NY             | Westchester County-NY | Wilkes-Barre/Scranton-PA |                    |
| Anchorage-AK     | Barrow-AK                | Bellingham-WA         | Bethel-AK                | Boise-ID           |
| Butte-MT         | Campbell River-BC        | Chico-CA              | Cordova-AK               | Dillingham-AK      |
| Eugene-OR        | Eureka/Arcata-CA         | Fairbanks-AK          | Fresno-CA                | Great Falls-MT     |
| Helena-MT        | Hilo-HI                  | Honolulu-HI           | Hoolehua-HI              | Juneau-AK          |
| Kahului-HI       | Kalispell-MT             | Kapalua-HI            | Kenai-AK                 | Ketchikan-AK       |
| Klamath Falls-OR | Kodiak-AK                | Kona-HI               | Kotzebue-AK              | Lanai City-HI      |
| Lewiston-ID      | Lihue-HI                 | Long Beach-CA         | Medford-OR               | Missoula-MT        |
| Modesto-CA       | Nanaimo-BC               | Nome-AK               | North Bend-OR            | Oakland-CA         |
| Pasco-WA         | Pendleton-OR             | Petersburg-AK         | Pocatello-ID             | Portland-OR        |
| Prince George-BC | Pullman-WA               | Quesnel-BC            | Redding-CA               | Redmond-OR         |
| Reno-NV          | Sacramento-CA            | San Francisco-CA      | San Jose-CA              | San Luis Obispo-CA |
| Sandspr-B-C      | <u>Seattle/Tacoma-WA</u> | Sitka-AK              | Smithers-BC              | Spokane-WA         |
| Valdez-AK        | Vancouver-BC             | Victoria-BC           | Walla Walla-WA           | Wenatchee-WA       |
| Williams Lake-BC | Yakima-WA                |                       |                          |                    |

(c)

FIGURE 3.25: AIR TRAVEL CLUSTERS BY AP

Albany-GA Asheville-NC Athens-GA Atlanta-GA Augusta-GA Birmingham-AL  
 Brunswick-GA Charleston-SC Charlotte-NC Chattanooga-TN Columbia-SC Columbus-GA  
 Columbus/Starkville/West Point-MS Dothan-AL Wilmington-NC Fayetteville-NC Florence-SC  
 Ft. Walton Beach-FL Greensboro/High Point/Winston Salem-NC Greenville-MS Valdosta-GA  
 Tupelo-MS Greenville/Spartanburg-SC Huntsville/Decatur-AL Jackson-MS Jacksonville-NC  
 Knoxville-TN Little Rock-AR Macon-GA Memphis-TN Montgomery-AL Muscle Shoals-AL  
 Myrtle Beach-SC Nashville-TN New Bern-NC Paducah-KY Savannah-GA Raleigh/Durham-NC  
 Roanoke-VA Springfield-MO Tri-City Airport-TN

---

Appleton-WI Bloomington-IL Brainerd-MN Cedar Rapids/Iowa City-IA Champaign-IL  
 Chicago-IL Dubuque-IA Duluth/Superior-MN Eau Claire-WI Fargo-ND Ft. Dodge-IA  
 Grand Rapids-MI Green Bay-WI Ironwood-MI Kalamazoo-MI La Crosse-WI Madison-WI  
 Manistee-MI Marquette-MI Mason City-IA Milwaukee-WI St. Cloud-MN Moline-IL  
 Muskegon-MI Peoria-IL Rhinelander-WI Rochester-MN Sioux Falls-SD South Bend-IN  
 Wausau-WI Traverse City-MI Minneapolis/St Paul-MN

---

Bakersfield-CA Chico-CA Crescent City-CA Eureka/Arcata-CA Fresno-CA Inyokern-CA  
 Las Vegas-NV Los Angeles-CA Modesto-CA Monterey-CA Oakland-CA Ontario-CA  
 Orange County-CA Oxnard/Ventura-CA Palm Springs-CA Redding-CA Reno-NV Sacramento-CA  
 San Diego-CA San Francisco-CA San Jose-CA San Luis Obispo-CA Santa Barbara-CA Santa Maria-CA

---

Albany-NY Allentown/Bethlehem/Easton-PA Altoona-PA Baltimore-MD Binghamton-NY  
 Charlottesville-VA Harrisburg-PA Hartford-CT New York-NY Philadelphia-PA Richmond-VA  
 Norfolk/Virginia Beach/Williamsburg-VA Salisbury-Ocean City-MD Shenandoah Valley-VA  
 State College-PA Washington-DC Wilkes-Barre/Scranton-PA

---

Anchorage-AK Barrow-AK Bethel-AK Cordova-AK Dillingham-AK Fairbanks-AK Juneau-AK  
 Kodiak-AK Kotzebue-AK Nome-AK Petersburg-AK Sitka-AK Valdez-AK Kenai-AK Ketchikan-AK

---

Charleston-WV Cincinnati-OH Columbus-OH Dayton-OH Detroit-MI Evansville-IN Flint-MI  
 Ft. Wayne-IN Indianapolis-IN Lansing-MI Lexington-KY Louisville-KY Saginaw-MI  
 Toledo-OH Huntington-WV

---

Bellingham-WA Campbell River-BC Comox-BC Eugene-OR Medford-OR Nanaimo-BC Portland-OR  
 Redmond-OR Vancouver-BC Victoria-BC Wenatchee-WA Yakima-WA Seattle/Tacoma-WA

---

Daytona Beach-FL Ft. Lauderdale-FL Ft. Myers-FL Gainesville-FL Jacksonville-FL Key West-FL  
 Melbourne-FL Miami-FL Orlando-FL Sarasota/Bradenton-FL Tallahassee-FL Tampa/St. Petersburg-FL  
 West Palm Beach-FL

---

Blanc Sablon-QC Charlottetown-PE Deer Lake-NL Fredericton-NB Goose Bay-NL Halifax-NS  
 Moncton-NB Sept-Îles-QC St. Anthony-NL St. John-NB St. Johns-NL Wabush-NL

---

Bradford-PA Clarksburg-WV Cleveland-OH Dubois-PA Erie-PA Franklin-PA Jamestown-NY  
 Johnstown-PA Parkersburg/Marietta-WV Pittsburgh-PA Morgantown-WV

FIGURE 3.26: KEY AIR TRAVEL CLUSTERS BY M1NN

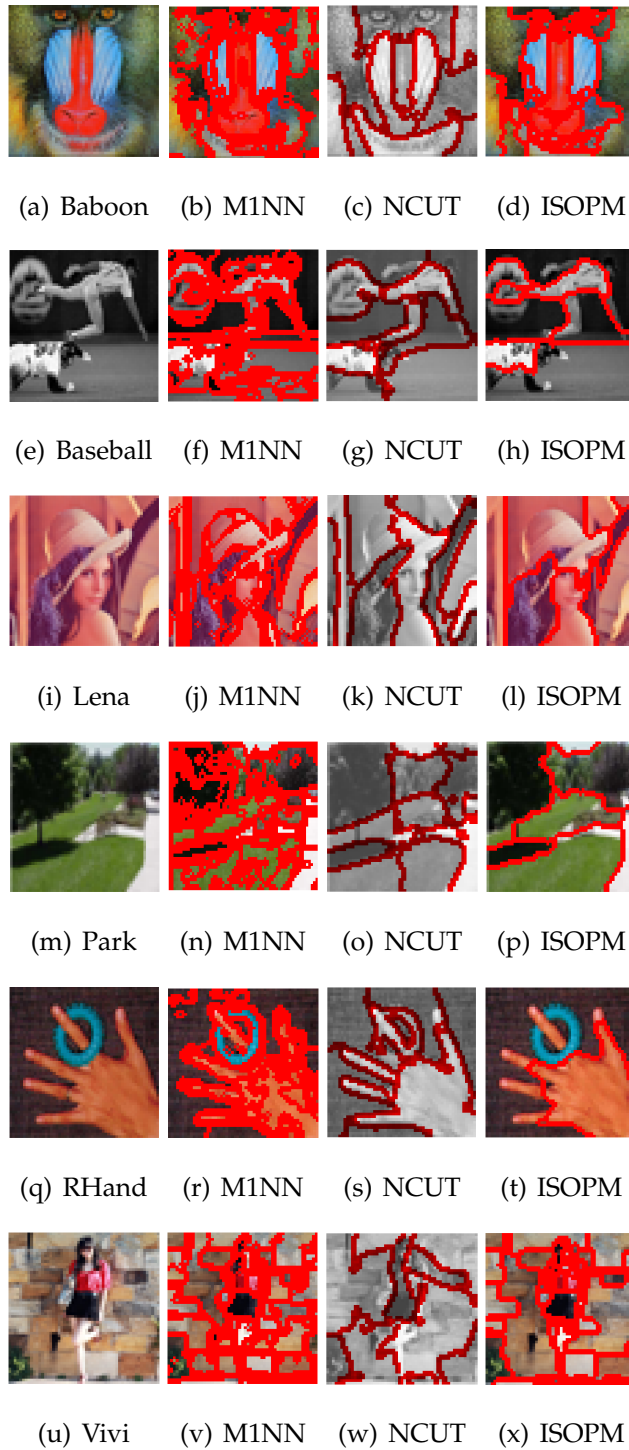


FIGURE 3.27: IMAGE SEGMENTATION COMPARISON



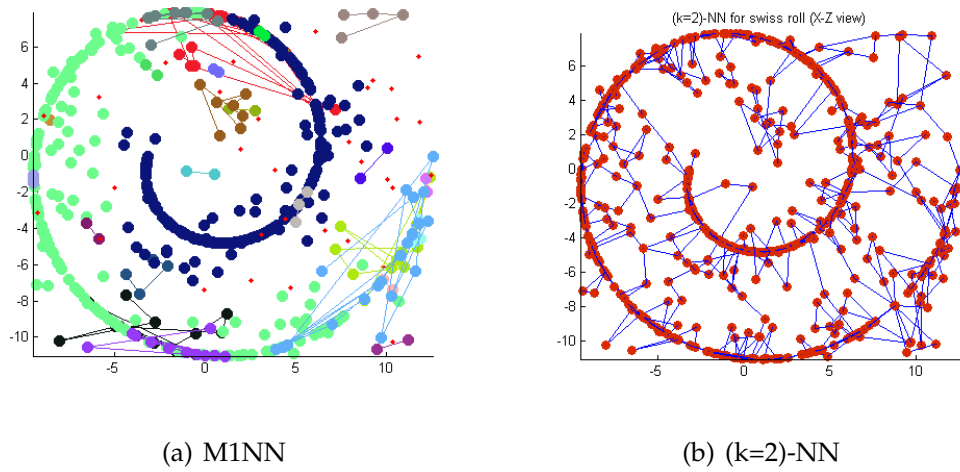


FIGURE 3.28: MANIFOLD LEARNING

correctly. In Figure 3.27(a), M1NN and ISOPM have quite similar segmentations. In Figure 3.27(e) and 3.27(m), M1NN gives a few redundant segmentations since it is not adequately optimized for image segmentation. In general, our method outperforms the other two methods with its promising potential demonstrated.

### 3.7.3 Manifold Learning

A manifold learning method is usually quite sensitive to noise, e.g. Isomap [166], as noise could remarkably affect the structure of a  $k$ -NN graph and the geodesic distance defined on the graph. To solve this problem, our method can be used to preprocess the data with heavy noise. By setting the cluster number  $k$  to 2 and discarding small clusters, the structure of the data can be retrieved from the remaining large clusters effectively. Figure 3.28 shows an example of learning a three dimensional Swiss roll (400 nodes) which is mixed with 200 noise nodes. In Figure 3.28(a), the spiral structure of the Swiss roll is preserved by the unconnected main clusters which are along the surface of the Swiss roll. On the other hand in Figure 3.28(b), the  $k$ -NN graph cannot work properly even with  $k = 2$

because of the '*short circuit*' problem. Thus the methods based on the  $k$ -NN graph are unable to capture the surface of the Swiss roll correctly.

### 3.8 Conclusion

In this chapter, we propose a new agglomerative clustering method based on the M1NN principle and parallel clustering algorithm. The experimental analysis demonstrates our method outperforms many other clustering methods. Our method can also be adapted for advanced applications, such as data mining, image segmentation and manifold learning.

For future work, There are three directions. First we want to investigate the strong merging rule for data sets in a non-Euclidean space. Second we want to improve our method for large scale data sets. Third we want to optimize our method for image segmentation.

## Chapter 4

# Modified Log-likelihood Clustering

In this chapter, we propose an intuitive MLL (Modified Log-Likelihood) model in the spirit of MDL for quantitative clustering analysis, which is developed on a probabilistic spectral framework [167]. The graph energy and log-likelihood function are adopted to define description lengths of clusterings for selecting the optimal clustering with the minimum description length.

### 4.1 MDL Principle for Clustering

MDL principle is a mathematical formalization of Occam's Razor. This principle asserts simpler theories are preferred until simplicity can be traded for greater explanatory power [168]. As reviewed in the previous chapter, MDL is a trade-off theory useful for choosing a specific hypothesis which minimizes the sum of the cost of coding the data by the hypothesis and the cost of coding the hypothesis itself. Considering the one-to-one correspondence between code length functions and probability distributions, MDL is viewed as equivalent to Bayesian inference by some researchers [68] (although non-Bayesian codes can also be accommodated [169]).

Clustering is the process of assigning a set of cluster labels  $\Omega = \{w_1, w_2, \dots, w_k\}$  to a set of objects  $X = \{x_1, x_2, \dots, x_n\}$ . Usually the preliminary knowledge of an adjacency matrix  $A$ , a corresponding graph  $G$  and an estimated cluster number  $k$  are provided. On an unweighted graph,  $A_{ij} = 1$  means there is an edge connecting node  $i$  and  $j$ , otherwise  $A_{ij} = 0$ . To represent the assignment between object  $i$  and cluster  $w$ , the cluster membership indicator  $s_{iw}$  is adopted. In hard clustering,  $s_{iw} = 1$  denotes object  $i$  is in cluster  $w$ , otherwise  $s_{iw} = 0$ . The vector of indicator variables for cluster  $w$  is given as  $s_w = (s_{1w}, s_{2w}, \dots, s_{nw})^T$ . The cluster membership matrix  $S = (s_1 | s_2 | \dots | s_k)$  can be used to indicate a hypothetical clustering from the hypothesis space  $\Theta$ , where  $s_k$  denotes the vector of indicators for the  $k$ -th cluster.

Based on the MDL principle and the ML (Maximum Likelihood) clustering framework developed by Robles-Kelly and Hancock [167], we define the optimal clustering  $S_{\text{MLL}}$  as

$$S_{\text{MLL}} = \arg \min_{S \in \Theta^*} \left\{ \beta \sum_{w \in \Omega} \sum_{i \in V_w} |\lambda_w(i)| - \sum_{w \in \Omega} \sum_{(i,j) \in \Phi} s_{iw} s_{jw} \ln \frac{A_{ij}}{1-A_{ij}} \right\} \quad (4.1)$$

where  $\Phi = V \times V - \{(i, i) | i \in V\}$  is the set of non-diagonal elements of  $A$  and  $V$  denotes the vertices of the graph  $G$ .  $\Theta^*$  denotes a subspace of clusterings which all contain the same number of clusters since our MLL model is intended to be used to compare the clusterings with equal number of clusters.  $\beta$  is the *inverse temperature* [170] and is set to 1 in experiments.  $\lambda_w(i)$  is the  $i$ -th eigenvalue of the adjacency matrix  $A_w$  of the *induced subgraph* [171] of  $G$  partitioned by the cluster  $w$  and  $V_w$  denotes the vertices of the induced subgraph. The term  $-\sum_{w \in \Omega} \sum_{(i,j) \in \Phi} s_{iw} s_{jw} \ln \frac{A_{ij}}{1-A_{ij}}$  measures the consistency of a clustering and the adjacency and  $\beta \sum_{w \in \Omega} \sum_{i \in V} |\lambda_w(i)|$  measures the complexity of the clustering.

Thus the description length function  $\mathcal{DL}(A, S)$  is defined as

$$\mathcal{DL}(A, S) = \beta \sum_{w \in \Omega} \sum_{i \in V_w} |\lambda_w(i)| - \sum_{w \in \Omega} \sum_{(i,j) \in \Phi} s_{iw} s_{jw} \ln \frac{A_{ij}}{1-A_{ij}} \quad (4.2)$$

### 4.1.1 Relationship to Log-likelihood Function

The pairwise clustering process was modeled by Robles-Kelly and Hancock as the outcome of a series of independent Bernoulli trials over all pairs of nodes, which is called a *Bernoulli process* [172]. Based on this model, they defined the log-likelihood function  $\mathcal{L}(A, S)$  [167]

$$\mathcal{L}(A, S) = \sum_{w \in \Omega} \sum_{(i,j) \in \Phi} \left\{ s_{iw} s_{jw} \ln \frac{A_{ij}}{1-A_{ij}} + \ln(1 - A_{ij}) \right\} \quad (4.3)$$

The log-likelihood function was maximized with respect to the adjacency  $A_{ij}$  and cluster membership variable  $s_{iw}$  in their iterative clustering method to refine the modal structure of the adjacency matrix. In our method, it is embedded to measure the consistency of a clustering and the adjacency. In the definition (4.3),  $\sum_{w \in \Omega} \sum_{(i,j) \in \Phi} \ln(1 - A_{ij}) = |\Omega| \sum_{(i,j) \in \Phi} \ln(1 - A_{ij})$  where  $|\Omega|$  denotes the number of clusters, is a constant term in our model as the clusterings to be compared consist of equal number of clusters, thus it is safely dropped from the description length function.

Classically, the association of the cluster  $w$  is defined as [2]

$$assoc(w) = \sum_{(i,j) \in \Phi} s_{iw} s_{jw} A_{ij} \quad (4.4)$$

From this aspect, the term  $-\sum_{w \in \Omega} \sum_{(i,j) \in \Phi} s_{iw} s_{jw} \ln \frac{A_{ij}}{1-A_{ij}}$  in our model corresponds to the negative sum of the individual cluster associations for the logarithmically transformed adjacency matrix  $\ln \frac{A_{ij}}{1-A_{ij}}$ . Minimizing this term is equivalent to maximizing the total associations, which is widely thought to work well with compact and well separated clusters [163].

### 4.1.2 Relationship to Graph Energy

The parameter cost of an MDL model is usually measured by an entropic quantity, which corresponds to the spectral term  $\beta \sum_{w \in \Omega} \sum_{i \in V_w} |\lambda_w(i)|$  in our model.

In mathematics, the sum of the absolute values of the eigenvalues of the adjacency matrix of a graph is called the energy of the graph [173]. More formally, let  $G$  be a simple graph of  $n$  vertices which does not contain loops or multiple edges and  $A$  the adjacency matrix of  $G$ , the eigenvalues of  $A$  are denoted by  $\lambda_i, i = 1, \dots, n$ , the graph energy is defined as

$$E(G) = \sum_{i=1}^n |\lambda_i| \quad (4.5)$$

A simple upper bound of graph energy was given as [174]

$$E(G) \leq \sqrt{2mn} \quad (4.6)$$

where  $m$  is the number of edges and  $n$  is the number of vertices.

A lower bound was proposed for graphs without isolated vertices [175]

$$E(G) \geq 2\sqrt{n-1} \quad (4.7)$$

with equality for star graphs. As can be seen, the energy of a graph is bounded by the number of edges and vertices of the graph. In theoretical chemistry, the  $\pi$ -electron energy of a conjugated carbon molecule coincides with the energy of its molecular graph [176]. The latest research reveals the connection between the energy and the dominating set of a graph [177]. The Laplacian energy of a graph [178] is even used in learning invariant structure for object identification [179].

In our method, the energy of a graph is used to define the complexity of the graph, which means a graph has lower complexity if the value of its energy is smaller. For example, Figure 4.1(a)-4.1(f) show the values of graph energy for a star graph, a line graph, a ring graph, a Delaunay graph, a  $k$ -NN graph with  $k = \lceil k_{\text{DG}} \rceil$  where  $k_{\text{DG}}$  is the average degree of the corresponding Delaunay graph and a complete graph, which all consist of 7 vertices. Based on our definition, in this example the star graph has the lowest complexity and the complete graph comes with the highest complexity.

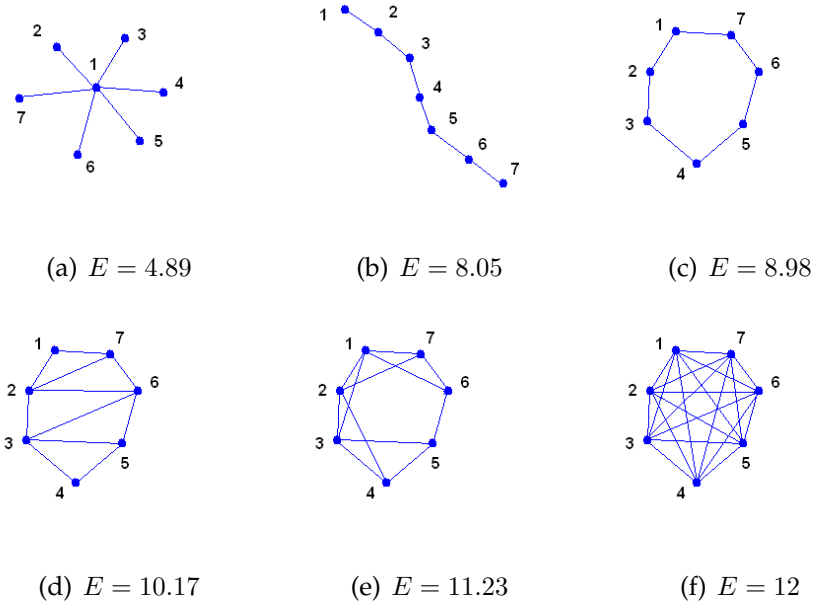
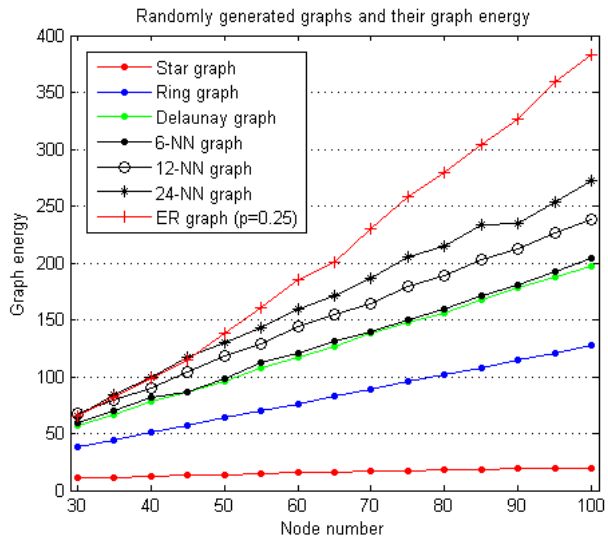
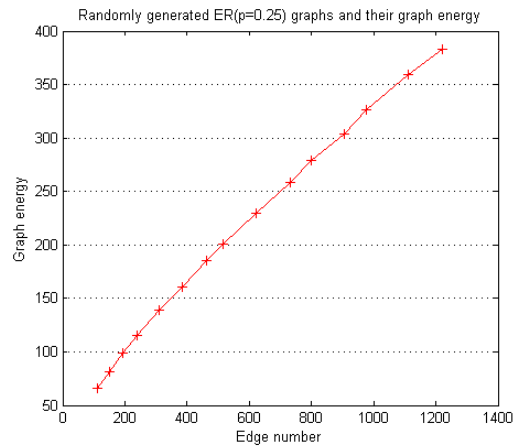
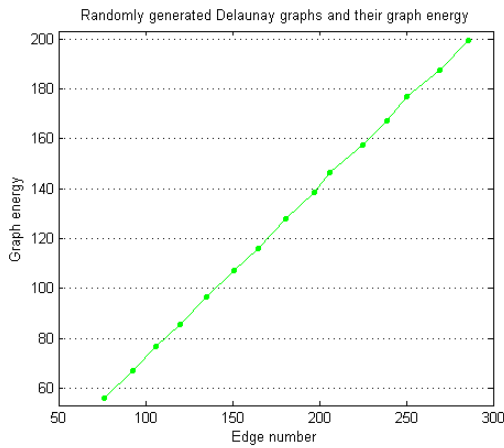


FIGURE 4.1: GRAPH ENERGY DEMONSTRATION

To further inspect the relationship between the energy and the number of nodes or edges of a graph, we construct star graphs, ring graphs, Delaunay graphs,  $k$ -NN graphs and ER random graphs [180] on the randomly generated nodes in the range of 30 to 100. As shown in Figure 4.2(a), the graph energy is monotonically proportional to the node number for all types of graphs. Also with the number of nodes increasing, ER random graphs have the highest energy and 24-NN graphs have the second highest energy whilst star graphs have the lowest energy. Since the edge number of a star graph is equal to its node number minus one, the edge number of a ring graph is equal to its node number and the edge number of a  $k$ -NN graph is approximately equal to its node number times  $k/2$ , Figure 4.2(b)-4.2(c) demonstrate only the relationships between graph energy and the edge number of Delaunay graphs and ER graphs, apparently which are highly linear as well. Thus it seems the graph energy, as a graph invariant, is closely related to the number of nodes and edges of each class of graphs.



(a) Graph energy versus node number



(b) Delaunay graph energy versus edge number      (c) ER graph energy versus edge number

FIGURE 4.2: GRAPH ENERGY INSPECTION



Intuitively, low complexity corresponds to simple graph structure, by simple we mean the graph may have less edges, vertices or cycles as shown in Figure 4.1 and 4.2. Thus the simplest clustering is considered as the graph partitioning which contains a number of simple subgraphs that the total complexity of the subgraphs is the lowest. Based on this idea,  $E_{s_w}$  as the energy of a cluster  $w$ , is defined by using the eigenvalues of  $A_w$  which is the corresponding principle submatrix of  $A$

$$E_{s_w} = \sum_{i=1}^{|V_w|} |\lambda_w(i)| \quad (4.8)$$

Hence  $\sum_{w \in \Omega} \sum_{i \in V_w} |\lambda_w(i)| = \sum_{w \in \Omega} E_{s_w} = E(S)$  denotes the total energy of the clustering  $S$ . Because clusters are complementary induced subgraphs of  $G$  in hard clustering, we have  $E(S) \leq E(G)$  [171], which implies the clustering with minimum total graph energy is the one that maximizes the loss of original graph energy.

## 4.2 Algorithm Description

The MLL clustering selection algorithm is given as:

1. Given the adjacency matrix  $A$  and the clustering space  $\Theta^* = \{S_1, \dots, S_m\}$ , compute the principle submatrix  $A_w$  for each cluster  $w$  of a clustering  $S$ .
2. Eigen-decompose  $A_w$  to have the eigenvalues  $\lambda_w$ .
3. Compute the description length  $\mathcal{DL}(A, S)$  for each clustering  $S$  using Equation (4.2).
4. Choose the optimal clustering  $S_{\text{MLL}}$  with the minimum  $\mathcal{DL}(A, S)$ .

## 4.3 Experimental Analysis

In this section, our clustering model is tested for graph partitioning selection and image segmentation comparison. The experiments on real data sets demonstrate that the optimal clusterings are correctly selected by using our model whilst the clustering methods can be quantitatively compared. Our model is also compared against the simpler ML (Maximum Likelihood) criterion on some clustering examples.

### 4.3.1 Toy Graph Partitioning Selection

In this section, different cuts [181] on a toy graph are quantitatively measured and compared. The graph partitionings are shown in Figure 4.3. The  $\mathcal{DL}$  values of the graph partitionings are  $\{685, 180, 108, 324, 180, 232\}$ . Apparently, graph partitioning 3 is considered optimal by our clustering model. Graph partitioning 1 is the poorest and is actually identical to the partitioning from the *minimum cut* [181]. Graph partitioning 2 and 5 are basically the same, which is detected by our method.

### 4.3.2 Clustering Selection

In this section, our method is applied to the clustering spaces generated by KM on the data sets 7G and York2012. There are 7 compact and well separated clusters contained in 7G, the clusterings of which are shown in Figure 4.4. The  $\mathcal{DL}$  scores are plotted in Figure 4.7. The optimal clusterings 1 and 3 are shown in Figure 4.4(a) and 4.4(c), which are actually the same and close to the true clustering. Similarly for York2012, the  $\mathcal{DL}$  scores are plotted in Figure 4.8. The optimal clusterings 5 and 8 are correctly detected as shown in Figure 4.5(e) and 4.5(h).

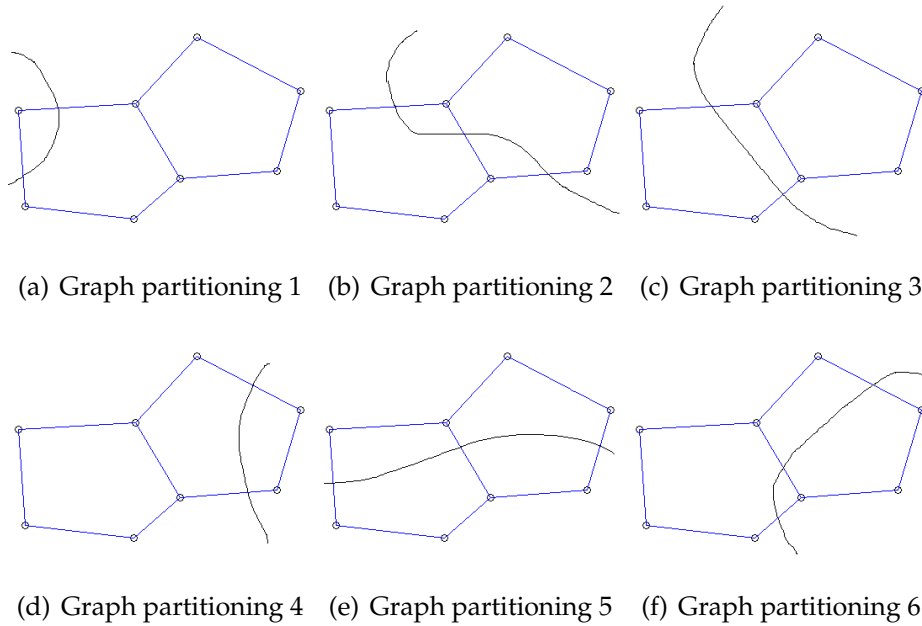


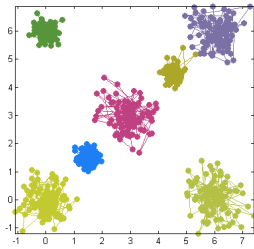
FIGURE 4.3: TOY GRAPH PARTITIONINGS

### 4.3.3 Image Segmentation Comparison

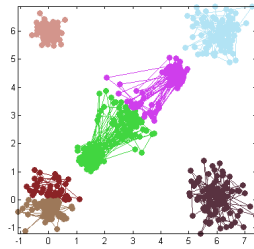
In this section, our method is applied to the image segmentations from NCUT by using the partitioning eigenvectors. The test image is from Figure 3.27(e) and is bipartitioned by 9 leading eigenvectors as shown in Figure 4.6. The  $\mathcal{DL}$  values are plotted in Figure 4.9. The optimal segmentations 7 and 9 are shown in Figure 4.6(g) and 4.6(i). Figure 4.6(b) shows the segmentation from the second smallest eigenvector, which has the third smallest  $\mathcal{DL}$  score. Since NCUT uses a recursive 2-way algorithm for image segmentation, our model may be helpful for NCUT to choose the proper eigenvectors with better performance.

### 4.3.4 Clustering Method Comparison

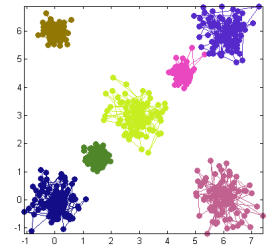
In this section, our method is applied to the clusterings generated from different clustering methods. The result is compared against GT (Ground Truth). The test data sets are from Table 3.1 and the  $\mathcal{DL}$  values are shown in Table 4.1. For each



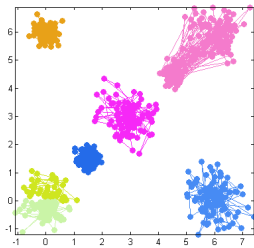
(a) KM clustering 1



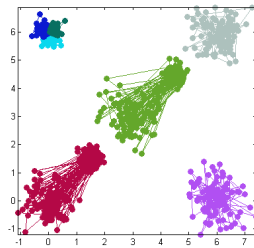
(b) KM clustering 2



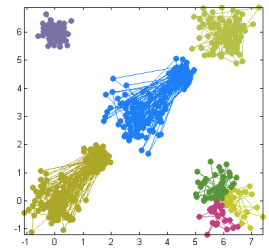
(c) KM clustering 3



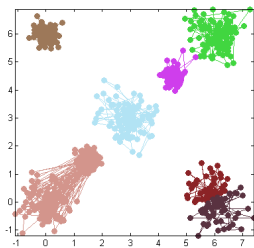
(d) KM clustering 4



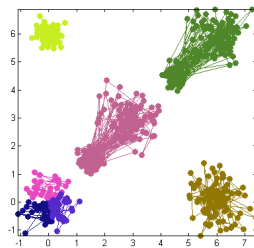
(e) KM clustering 5



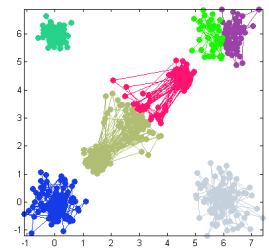
(f) KM clustering 6



(g) KM clustering 7

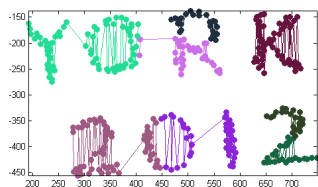


(h) KM clustering 8

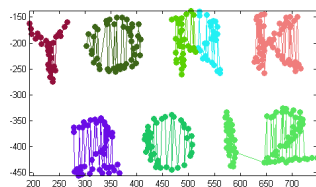


(i) KM clustering 9

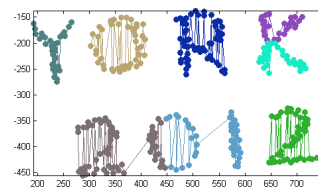
FIGURE 4.4: KM CLUSTERINGS ON 7G (7 GAUSSIAN CLUSTERS)



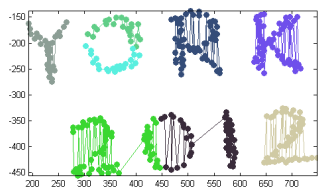
(a) KM clustering 1



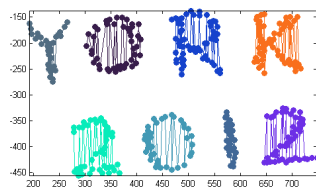
(b) KM clustering 2



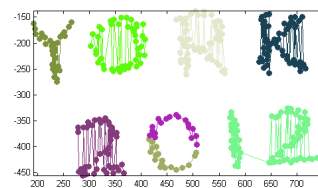
(c) KM clustering 3



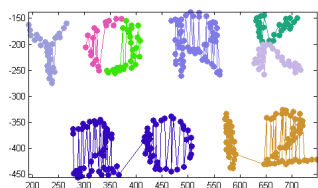
(d) KM clustering 4



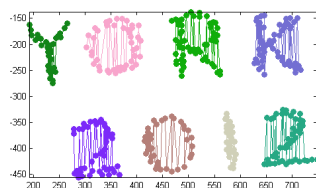
(e) KM clustering 5



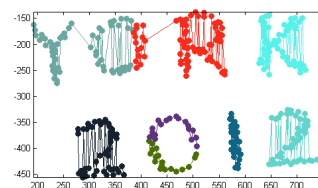
(f) KM clustering 6



(g) KM clustering 7

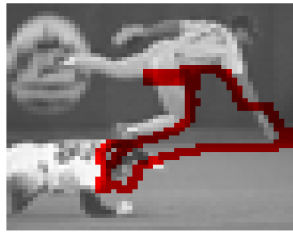


(h) KM clustering 8

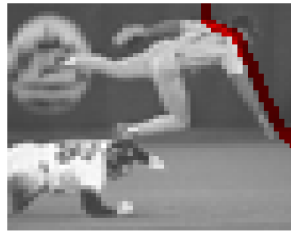


(i) KM clustering 9

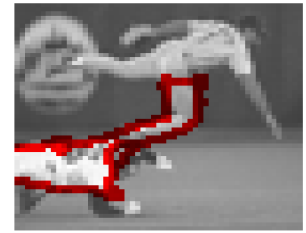
FIGURE 4.5: KM CLUSTERINGS ON YORK2012



(a) Image segmentation 1



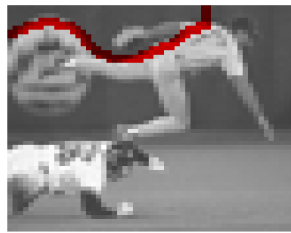
(b) Image segmentation 2



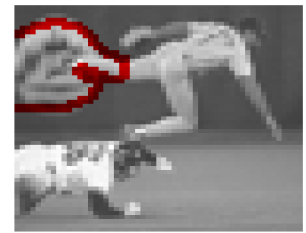
(c) Image segmentation 3



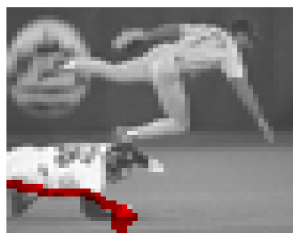
(d) Image segmentation 4



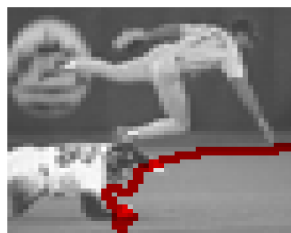
(e) Image segmentation 5



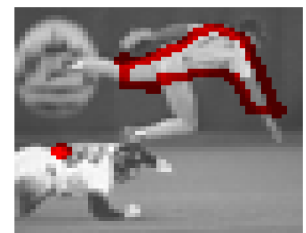
(f) Image segmentation 6



(g) Image segmentation 7



(h) Image segmentation 8



(i) Image segmentation 9

FIGURE 4.6: TOY IMAGE SEGMENTATION

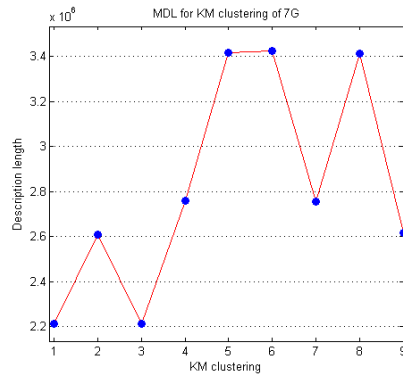


FIGURE 4.7: DL OF KM CLUSTERINGS ON 7G

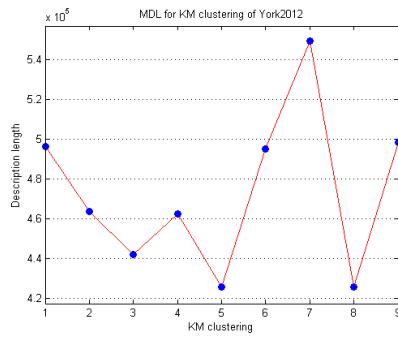


FIGURE 4.8: DL OF KM CLUSTERINGS ON YORK2012

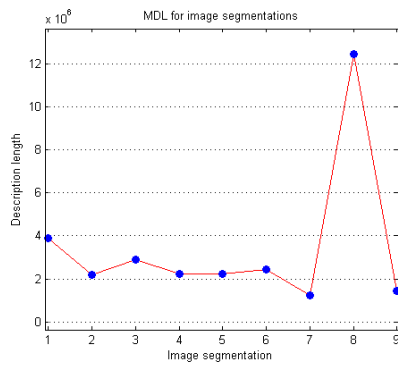


FIGURE 4.9: DL OF IMAGE SEGMENTATIONS

data set, the methods which could create sparse clusterings are not compared as our clustering model is based on assuming clusterings are comprised of the same number of clusters. Apparently, M1NN has the best performance overall. NCUT is generally better than AP and KM. HC performs very unstably and has extreme results in many cases, which has been observed previously.

### 4.3.5 Further Discussion on MLL and ML

The ML framework developed by Robles-Kelly and Hancock is adapted to a more complicated form, namely the MLL model, in which one part corresponds to the negative log-likelihood which measures the total associations for the logarithmically transformed adjacency matrix, and the other part measures the complexity of a clustering in terms of graph energy. On the other hand, the MLL model can only be used to compare the clusterings with equal number of clusters, which limits its usefulness for clustering analysis.

Nevertheless in practice, it is found these two criteria behave very similarly in most experiments as shown in Table 4.1 and 4.2. However in some circumstances, the MLL model can make a decision by the additional information whilst ML cannot. For example in Figure 4.10(a) and 4.10(b), a toy graph is bipartitioned by Cut1 and Cut2. Both MLL and ML choose the optimal partitioning S2. But when node 11 is further connected to 14 and 19 as shown in Figure 4.11, ML cannot select the optimal partitioning between S1 and S2 since they have the same score as shown in Table 4.3. In the meantime, MLL can still make the choice of picking the partitioning S1 in Figure 4.11(a), which appears to be appropriate as there are relatively more links between node 11 and the smaller cluster.



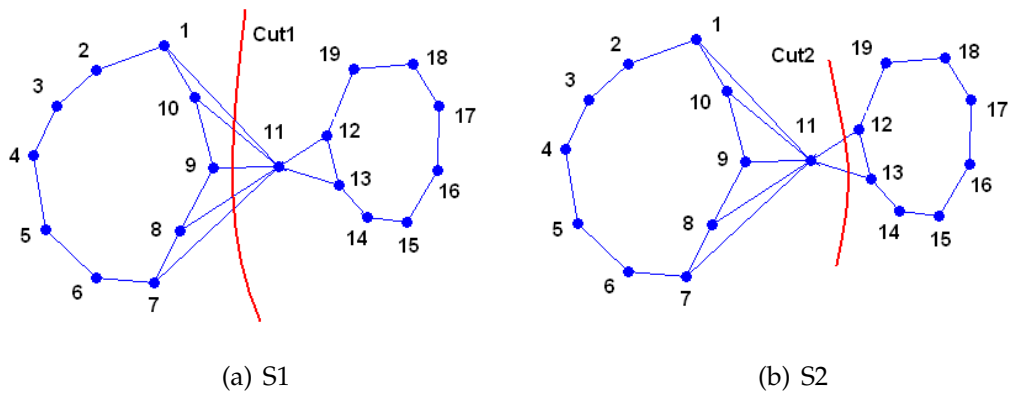


FIGURE 4.10: BIPARTITIONED TOY GRAPH 1

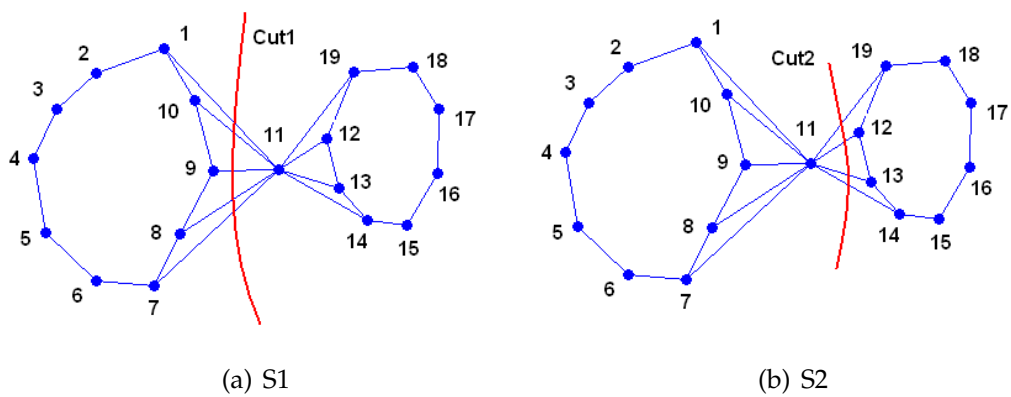


FIGURE 4.11: BIPARTITIONED TOY GRAPH 2

TABLE 4.1: DESCRIPTION LENGTH FOR CLUSTERING METHODS

Unit:  $\times 10^5$ nit

|          | M1NN  | KM    | AP    | NCUT  | ISOPM | HC     | CHA   | GT    |
|----------|-------|-------|-------|-------|-------|--------|-------|-------|
| Caltech2 | -     | 12.82 | 12.88 | 13.16 | -     | 28.98  | -     | 9.72  |
| Caltech3 | 7.82  | 8.19  | 8.28  | 7.82  | -     | 7.82   | -     | 7.82  |
| Caltech4 | -     | 30.93 | 30.15 | 25.95 | -     | 129.14 | -     | 25.64 |
| Caltech5 | 21.62 | 26.24 | 26.07 | 21.62 | -     | 21.62  | -     | 21.62 |
| Caltech6 | -     | 8.14  | 8.14  | 6.88  | -     | 8.96   | 6.23  | 6.18  |
| 2S       | 26.16 | 26.17 | 26.17 | 29.39 | 39.10 | 26.16  | 44.62 | 26.16 |

TABLE 4.2: LOG-LIKELIHOOD FOR CLUSTERING METHODS

Unit:  $\times 10^5$ nit

|          | M1NN   | KM     | AP     | NCUT   | ISOPM  | HC      | CHA    | GT     |
|----------|--------|--------|--------|--------|--------|---------|--------|--------|
| Caltech2 | -      | -14.75 | -14.80 | -15.09 | -      | -30.91  | -      | -11.65 |
| Caltech3 | -9.50  | -9.87  | -9.95  | -9.50  | -      | -9.50   | -      | -9.50  |
| Caltech4 | -      | -37.59 | -36.98 | -36.81 | -      | -135.80 | -      | -32.29 |
| Caltech5 | -26.00 | -30.62 | -30.45 | -26.00 | -      | -26.00  | -      | -26.00 |
| Caltech6 | -      | -9.62  | -9.62  | -8.36  | -      | -10.44  | -7.71  | -7.66  |
| 2S       | -26.71 | -26.72 | -26.72 | -29.95 | -39.65 | -26.71  | -45.17 | -26.71 |

TABLE 4.3: MLL AND ML COMPARISON

|                | S1             | S2             | S1             | S2             |
|----------------|----------------|----------------|----------------|----------------|
|                | Figure 4.10(a) | Figure 4.10(b) | Figure 4.11(a) | Figure 4.11(b) |
| $\mathcal{DL}$ | 2981           | 2693           | 2692           | 2693           |
| $\mathcal{L}$  | -6560          | -6272          | -6560          | -6560          |

## 4.4 Conclusion

In this chapter, we propose a modified log-likelihood model in the spirit of MDL to select the optimal clustering based on its description length. The experimental analysis demonstrates our method is able to correctly detect such an optimal clustering from the clustering space. Our method can also be used to select proper eigenvectors for efficient image segmentation and quantitatively compare clustering methods.

For future work, There are two directions. First we want to improve this model for overlapped clusterings and relaxed cluster membership values. Second we are interested to develop a feature selection algorithm based on our method.

## Chapter 5

# Delaunay Graph Characterization and Graph-Based Image Matching

In this chapter, graph characterization methods are investigated for characterizing Delaunay graphs. The normalized Euclidean distance formulated from heat diffusion on a Delaunay graph is found particularly useful to estimate pairwise distances. A graph-based matching method is proposed for images under slight translation, scaling and rotation.

### 5.1 Heat Diffusion on Delaunay Graphs

In the literature, the heat kernel was used to approximate geodesic distances between nodes but its performance dropped rapidly when a critical value of time was exceeded [120]. HKS (Heat Kernel Signature), as a vector containing the trace of the heat kernel matrix, can only work properly with correct time parameter [182]. In this section, the critical time  $t_c$  is proposed as a measure of the diameter of a Delaunay graph. We use  $t_c$  to calculate the heat kernel and estimate pairwise distances. This method is proved more accurate than commute time.

### 5.1.1 Critical Time of Heat Diffusion

The concept of time in the heat kernel was synonymously expressed as the inverse temperature  $\beta$  from the *Estrada index* [183, 184, 185]. When  $\beta \rightarrow 0$ , interaction strength tends to zero and a graph is disconnected to own a 'gas' like state. When  $\beta \rightarrow \infty$ , the graph is most 'solid' at the lowest temperature.

The heat diffusion was also used to define the *thermodynamic depth* [186] for characterizing network complexity. The depth of a graph relies on the variability of causal trajectories, which are characterized by the heat flow complexity  $\mathcal{F}_\beta(G)$ . When  $\beta \rightarrow 0$ ,  $\mathcal{F}_\beta(G) = 0$ . When  $\beta \rightarrow \infty$ , it corresponds to the equilibrium state and  $\mathcal{F}_\beta(G) = 1$ .

The heat kernel can also be expressed with a Poisson distribution [187, 188], where time is the expected number of steps for all random walks with a certain length. This can be seen by rewriting the heat kernel

$$H_t(u, v) = \sum_{k=0}^{\infty} \frac{e^{-t} t^k}{k!} \mathcal{A}^k(u, v) \quad (5.1)$$

where  $\mathcal{A}^k(u, v)$  is the sum of all random walks of length  $k$  from node  $u$  to node  $v$  on a graph  $\mathcal{G}$ . In this expression, the weighting for each  $\mathcal{A}^k(u, v)$  is adjusted by the Poisson distribution. A small time will favor short walks and the localized commuting also presents a 'gas' like graph.

Thus it seems a critical time should be neither too small nor too big to characterize a Delaunay graph. In the literature, Euclidean graphs were approximated by unweighted Delaunay graphs [189, 190, 191] for solving the *geometric spanner* problem [192]. If the pairwise distances on a Delaunay graph can be well estimated by heat diffusion at a certain time, that time is considered critical for Delaunay graph characterization.

The critical time  $t_c$  is defined as

$$t_c = \min \left\{ k \mid \forall A^k(i, j) \neq 0, i \neq j \right\} \quad (5.2)$$

where  $A^k(i, j)$  is the  $(i, j)$ -th element of the matrix  $A^k$ . In other words, a Delaunay graph is converted by  $t_c$  to a weighted complete graph that any two nodes are directly connected.

### 5.1.2 Delaunay Graph Characterization by Heat Diffusion

A Delaunay graph is usually shown on top of an object which brings the illusion that shape information or pairwise distances between nodes are contained in the Delaunay graph. In the literature, effort has been made to roughly estimate pairwise distances for a Delaunay graph [118, 193]. In this section, the distances computed from heat diffusion at different times are investigated for Delaunay graph characterization.

The geodesic distance of node  $x$  and  $y$  is formulated from heat diffusion as [194, 195]

$$d_G^2(x, y) = -4t \ln \left\{ (4\pi t)^{\frac{n}{2}} H_t(x, y) \right\} \quad (5.3)$$

where  $t$  is the time parameter and  $n$  is the dimension of a data space. For a planar Delaunay graph,  $n = 2$ . The Euclidean distance between two nodes on a graph plane can be approximated by their geodesic distance [196, 197]

$$d_E^2(x, y) \cong d_G^2(x, y) = -4t \ln \left\{ 4\pi t H_t(x, y) \right\} \quad (5.4)$$

where  $t$  needs to be small enough. Also the term  $4\pi t H_t(x, y)$  inside the logarithm needs to be between 0 and 1 for the squared distance to be meaningful. Therefore we introduce a normalization step

$$d_{EN}^2(x, y) = \frac{d_G^2(x, y) - d_{G\min}^2}{d_{G\max}^2 - d_{G\min}^2} = \frac{\ln\{H_t(x, y)/H_{t\max}\}}{\ln\{H_{t\min}/H_{t\max}\}} \quad (5.5)$$

In this way the logarithmic term is always negative and the normalized distance  $d_{EN}^2(x, y) \in [0, 1]$  all the time. Essentially this relative distance measure is fully dependent on the ratio of heat kernel values, of which both the maximum and

minimum are taken as references. The advantage of using this distance measure instead of the graph geodesic which was defined as the shortest path on a graph [198] is quite obvious that the distance from heat diffusion does not rely on just the shortest path therefore is more robust and efficient to use.

To see if the normalized Euclidean distance is useful for Delaunay graph characterization, firstly a Delaunay graph is built for the data set NC1 from Table 3.1. Figure 5.1(a) shows the sparse adjacency matrix. Figure 5.1(b) shows the distance matrix which has a clear block structure. The distance matrix is computed from the spatial coordinates of data points in the Euclidean space and is regarded as the ground truth. In Figure 5.1(c), the commute time matrix has a weakly visible block structure. Figures from 5.1(d) to 5.1(i) demonstrate the normalized distance matrices which are computed from the heat kernel with increasing time. Clearly the distance matrix from the critical time is the best approximation to the ground truth as shown in Figure 5.1(f). It outperforms the commute time matrix and the distance matrices from other times.

Secondly, a more sophisticated data set Complex8 is tested, which is also from Table 3.1. The commute time matrix is shown in Figure 5.2(c), which presents the clusters of the data set in vaguely apparent blocks. The normalized distance matrices are shown from Figure 5.2(d) to 5.2(i). Comparing with the ground truth in Figure 5.2(b), the normalized distance matrix from the critical time is still the best approximation and evidently beats the commute time matrix. In the meantime, Complex8 is found to own a larger critical time than NC1 because there are much more data points contained in it. The critical times for the data sets from Table 3.1 are summarized in Table 5.1.

In general, as demonstrated by these experiments, the pairwise distance on a Delaunay graph can be effectively estimated by the normalized Euclidean distance at the critical time, which is more accurate than commute time.

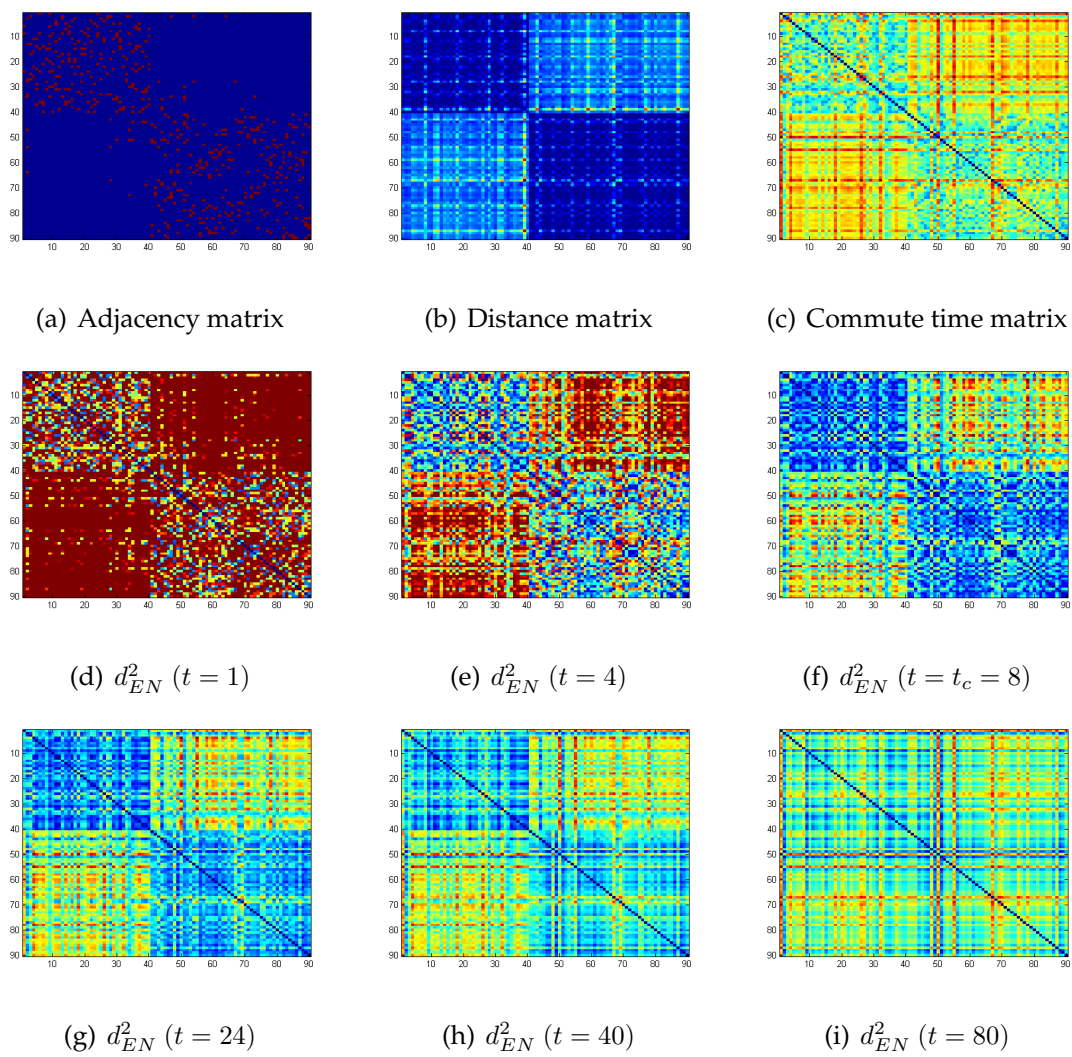
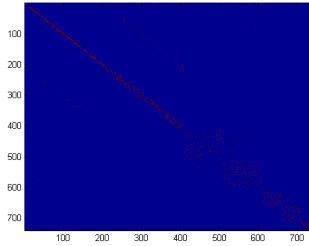
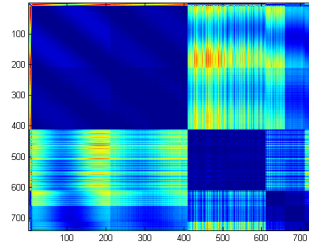


FIGURE 5.1: THE DISTANCE MATRICES OF NC1

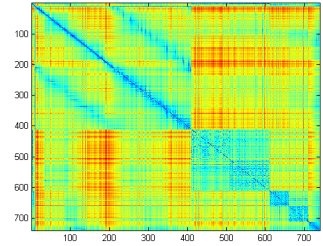




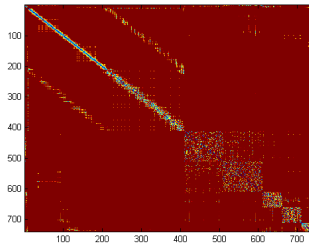
(a) Adjacency matrix



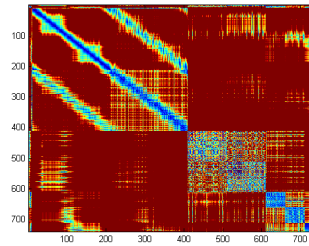
(b) Distance matrix



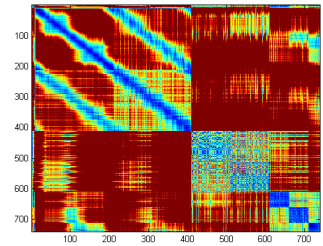
(c) Commute time matrix



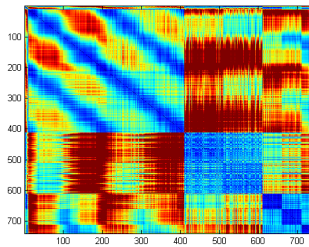
(d)  $d_{EN}^2(t=1)$



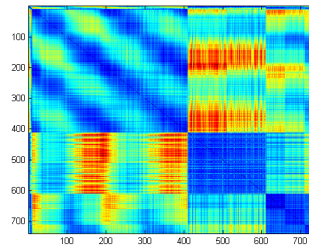
(e)  $d_{EN}^2(t=8)$



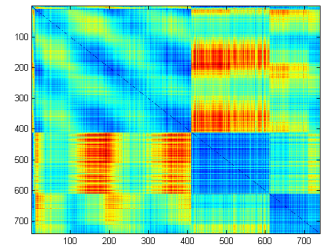
(f)  $d_{EN}^2(t=t_c=16)$



(g)  $d_{EN}^2(t=40)$



(h)  $d_{EN}^2(t=88)$



(i)  $d_{EN}^2(t=160)$

FIGURE 5.2: THE DISTANCE MATRICES OF COMPLEX8

TABLE 5.1: CRITICAL TIMES FOR THE DELAUNAY GRAPHS

| Data     | Graph node number | Critical time |
|----------|-------------------|---------------|
| NC1      | 90                | 8             |
| NC2      | 100               | 8             |
| NC3      | 130               | 9             |
| Caltech1 | 299               | 13            |
| Caltech2 | 303               | 14            |
| Caltech3 | 266               | 11            |
| Caltech4 | 622               | 16            |
| Caltech5 | 512               | 13            |
| Caltech6 | 238               | 14            |
| Complex8 | 739               | 16            |
| 2S       | 386               | 14            |
| RD       | 600               | 16            |
| RG       | 290               | 12            |

## 5.2 Graph-Based Image Matching

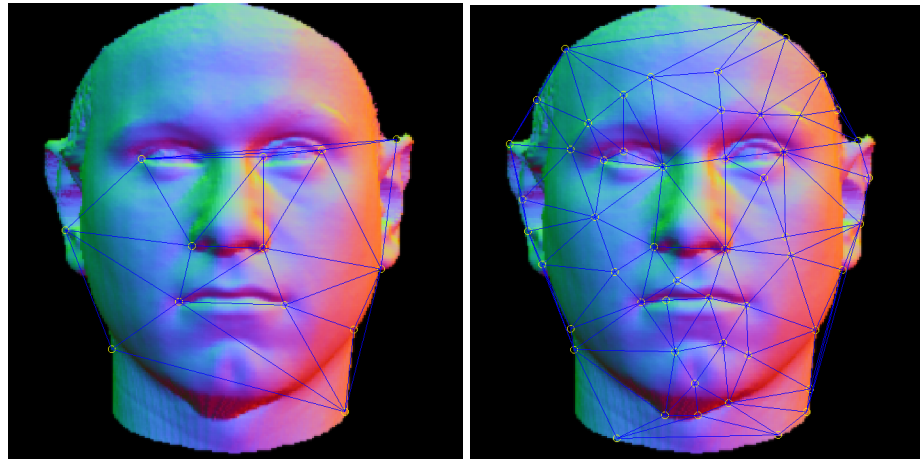
Graph-based image matching could be more robust and accurate than point pattern (pixel) matching [199] since the graph representation can provide additional information about neighborhood for each feature point. In this section, we propose a graph-based image matching method, which uses the result of diffusion process on a Delaunay graph studied previously.

### 5.2.1 Delaunay Graph Selection for Image Matching

When parameters are set, a Harris corner detector [137] can automatically find a number of feature points on an image. These points become the vertices of a Delaunay graph for matching. But the parameters cannot guarantee there are enough vertices to properly represent the image, or there may be much more vertices than needed. Conventionally the parameters are manually adjusted to select the best graph representation.

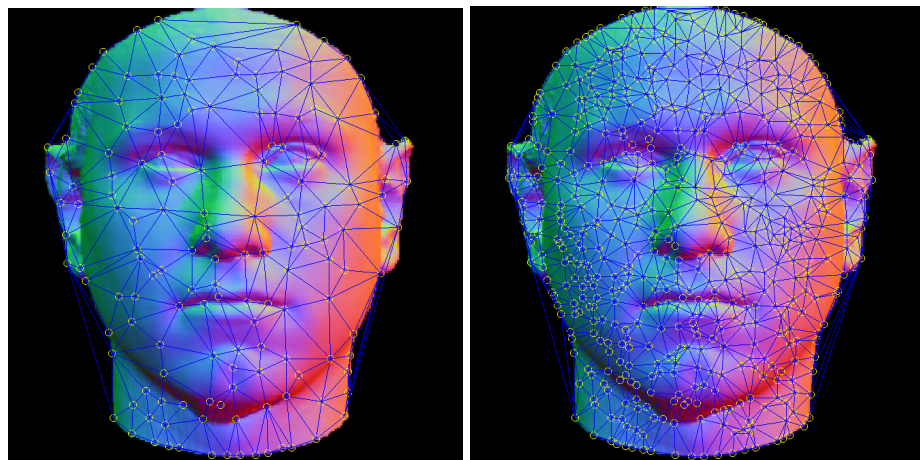
On the other hand, the critical time  $t_c$  as a meaningful measure for Delaunay graphs, can be directly used for graph selection. In Figure 5.3, Delaunay graphs with various  $t_c$  are plotted on a face image [200]. Figure 5.3(a) shows when  $t_c$  is small, the graph cannot adequately cover the face. Figure 5.3(d) demonstrates the graph with large  $t_c$  has too many redundant feature points, which is overly descriptive for the face and results an increase in computational cost. For this face image, Figure 5.3(b) is the most appropriate graph.

To investigate if there is a  $t_c$  generally suitable for any image of normal size, e.g.  $300 \times 300$  pixels, experiments are run on image databases including the COIL-20 toys [201], the Max Planck faces [200], the CMU house sequences [202], etc. It is found a Delaunay graph with  $t_c = 6$  or  $7$  can optimally represent an image. Thus we set  $t_c = 7$  in our image matching method.



(a)  $t_c = 3$

(b)  $t_c = 7$



(c)  $t_c = 13$

(d)  $t_c = 21$

FIGURE 5.3: DELAUNAY GRAPHS WITH DIFFERENT  $t_c$  ON A FACE IMAGE

In practice, the Delaunay graph with  $t_c = 7$  can be generated automatically by initially setting the parameter of non-maximal suppression of the Harris corner detector to a lower value, triangulating the feature points and computing the  $t_c$  of the graph, then increasing the value of the parameter accordingly. Normally this process takes 3 to 4 iterations in our experiments.

## 5.2.2 Graph Node Selection for Image Matching

Inexact graph matching methods, e.g. the random walks [203] or discrete relaxation [204], had been used for graph-based image matching. However a slight affine transformation on an image plane could change the local graph structure radically, in which case graph matching methods may not match images correctly. To solve this problem, we propose a method to select the *hot nodes* of graphs. The hot nodes are then matched by their feature descriptors.

Hot nodes are defined based on heat diffusion on a Delaunay graph. Given a graph with  $m$  vertices, the heat kernel matrix  $H_t$  has  $m^2$  entries. The hot node is defined as

$$n_h = \{i | H_t(i, i) > \mu\} \quad (5.6)$$

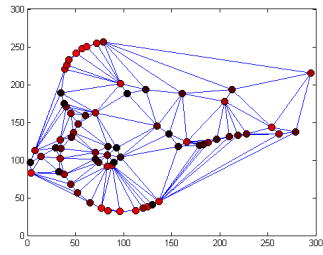
where  $\mu$  is a threshold, which is set equal to the  $m + 1$ -th maximum entry of  $H_t$  in our work. In other words, we select the first  $m$  biggest entries from the heat kernel matrix, which may include nodes and edges. Then we further select the nodes from these entries.

Figure 5.4 shows the Delaunay graphs for an image of a duck toy [201]. The heat of hot nodes is visualized by red color, e.g. hottest nodes are marked lightest red. When  $t = 1$ , all nodes are hot as shown in Figure 5.4(a). When  $t = 25$ , there are only 8 hot nodes left. By experiment, it is found the critical time  $t_c$  is useful for selecting a proper number of stable nodes on a graph. This is demonstrated in Figure 5.5 for a sequence of duck images. For any two sequential images, e.g.

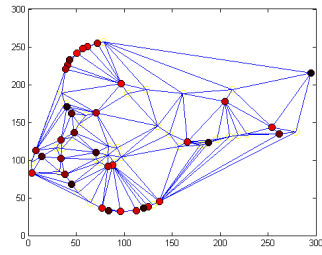
Figure 5.5(d) and 5.5(e), the correspondence can always be set up between some stable hot nodes even the graph structure is altered mildly.

To demonstrate the nodes selected by our method are stable and reliable for matching, a comparison of our method against centrality measures is presented on the same graphs. For each graph, 20% of total nodes with the largest centralities are chosen and highlighted in red color as shown in Figure 5.6-5.9. As can be seen, our method outperforms centrality measures marginally by selecting more useful nodes for matching. For example, there are 7 nodes to match between Figure 5.6(d) and 5.6(e) by degree centrality, 8 nodes between Figure 5.7(d) and 5.7(e) by closeness centrality, 8 nodes between Figure 5.8(d) and 5.8(e) by betweenness centrality, 6 nodes between Figure 5.9(d) and 5.9(e) by eigenvector centrality and 9 nodes between Figure 5.5(d) and 5.5(e) by our method. For the full COIL-20 database, our method has a better overall performance.

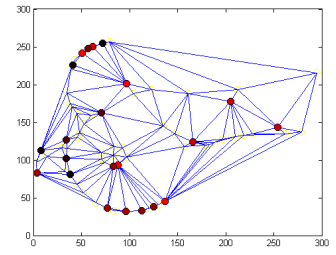
The dominant set partitioning method [63] is also tested to investigate whether graph partitions can be used for node selection. The maximum iteration is set to 1000 and selected nodes are marked red. As shown in Figure 5.10, the node selection process is, to a great extent, affected by the structural variations in graphs. For example, there are 5 nodes valid for matching between Figure 5.10(a) and 5.10(c), on the other hand there are 8 nodes between Figure 5.6(a) and 5.6(c) by degree centrality, 7 nodes between Figure 5.7(a) and 5.7(c) by closeness centrality, 9 nodes between Figure 5.8(a) and 5.8(c) by betweenness centrality, 7 nodes between Figure 5.9(a) and 5.9(c) by eigenvector centrality and 10 nodes between Figure 5.5(a) and 5.5(c) by our method. Apparently the dominant set partitioning method may not work properly for node selection since it is not designed for such a purpose. For the full COIL-20 database, the performance of this method cannot match ours.



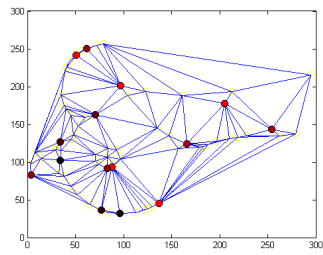
(a)  $t = 1$



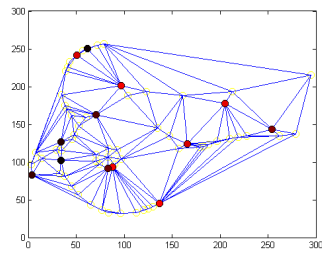
(b)  $t = 4$



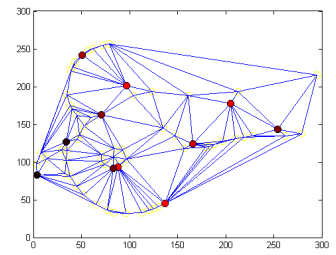
(c)  $t = 7$



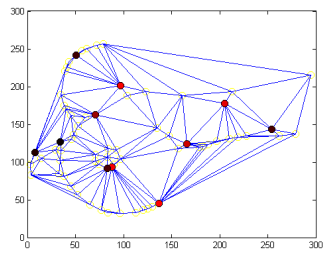
(d)  $t = 10$



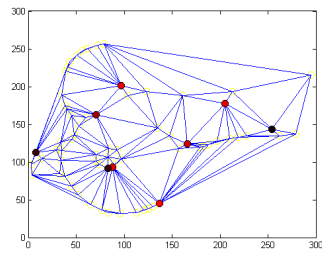
(e)  $t = 13$



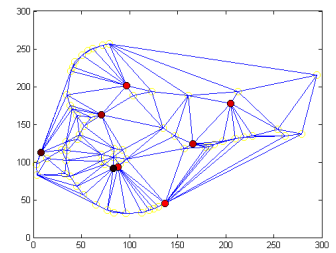
(f)  $t = 16$



(g)  $t = 19$

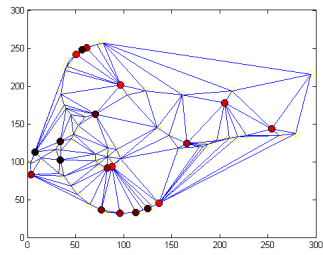


(h)  $t = 22$

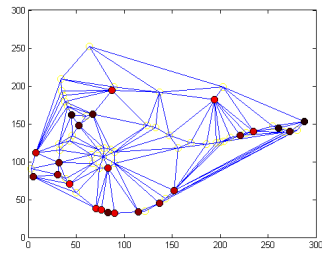


(i)  $t = 25$

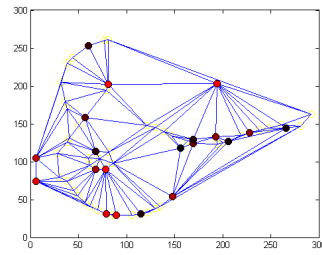
FIGURE 5.4: HOT NODE EVOLUTION ON A GRAPH BY INCREASING TIME



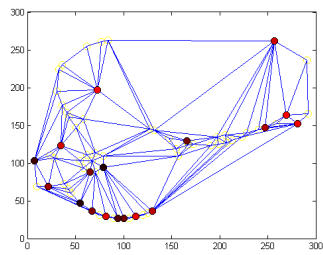
(a) Img1



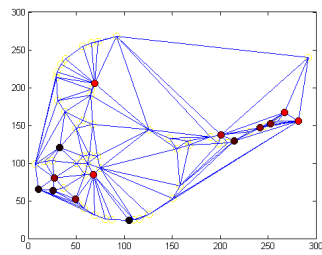
(b) Img2



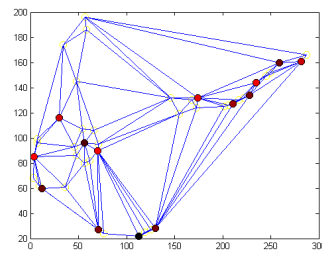
(c) Img3



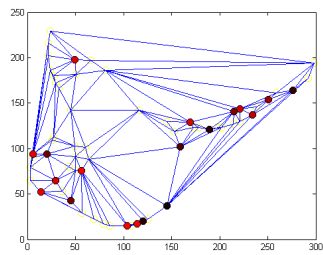
(d) Img4



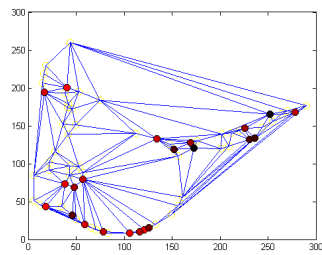
(e) Img5



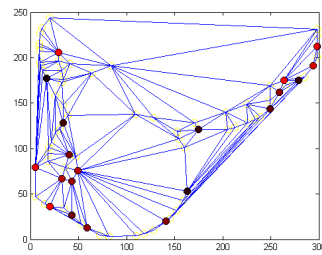
(f) Img6



(g) Img7



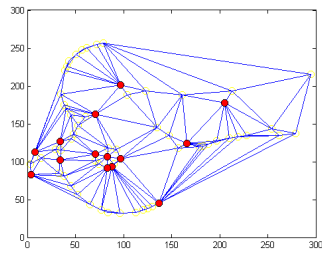
(h) Img8



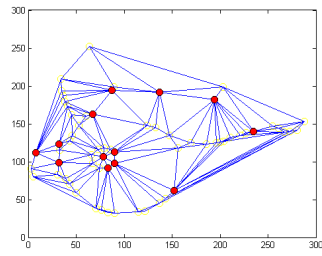
(i) Img9

FIGURE 5.5: HOT NODE SELECTION BY USING HEAT DIFFUSION AT  $t_c$

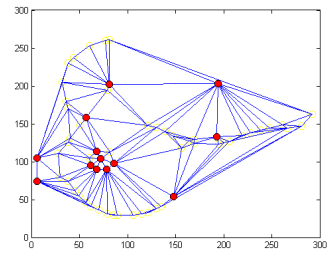




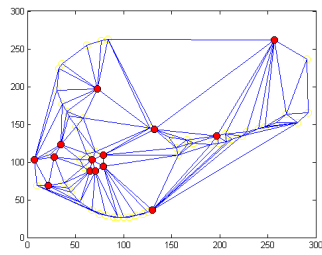
(a) Img1



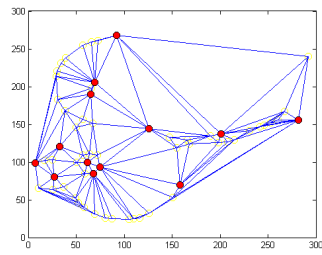
(b) Img2



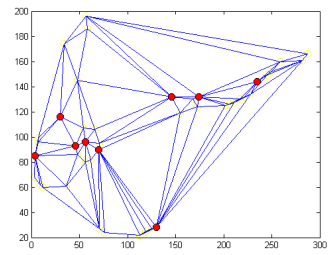
(c) Img3



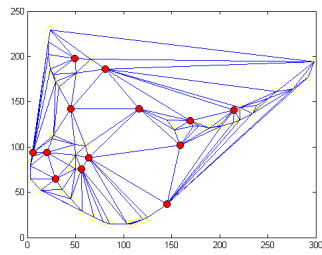
(d) Img4



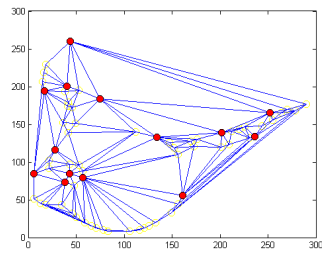
(e) Img5



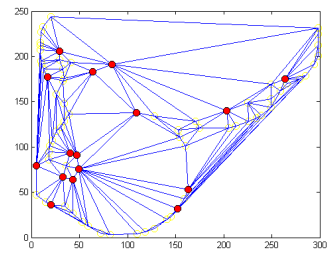
(f) Img6



(g) Img7

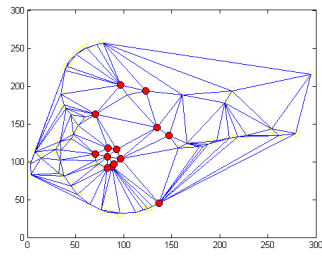


(h) Img8

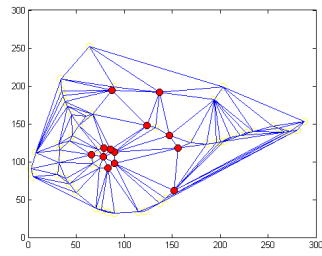


(i) Img9

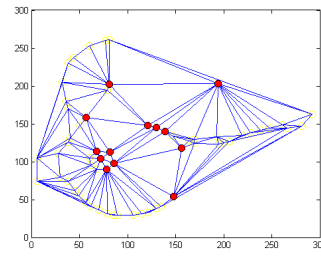
FIGURE 5.6: NODE SELECTION BY USING DEGREE CENTRALITY



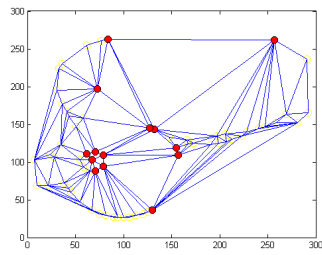
(a) Img1



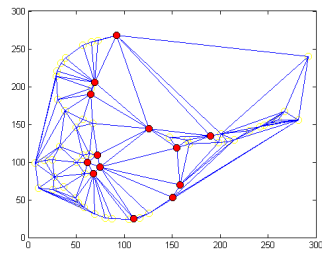
(b) Img2



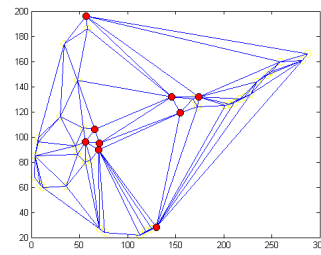
(c) Img3



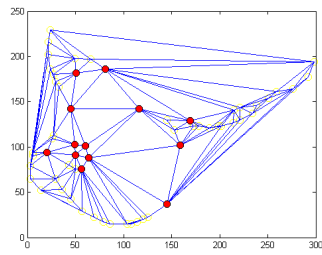
(d) Img4



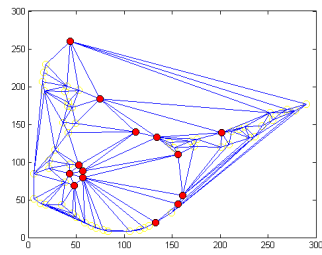
(e) Img5



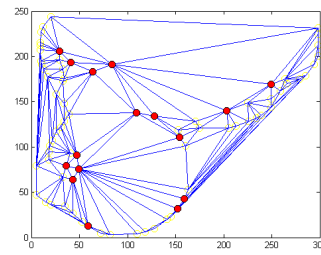
(f) Img6



(g) Img7

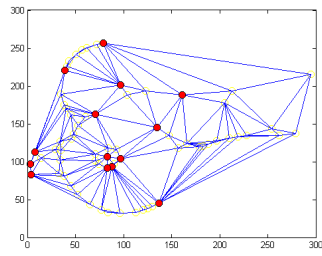


(h) Img8

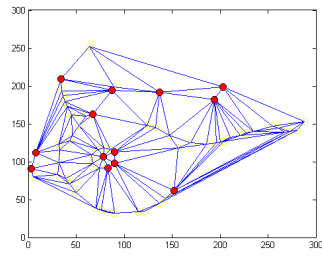


(i) Img9

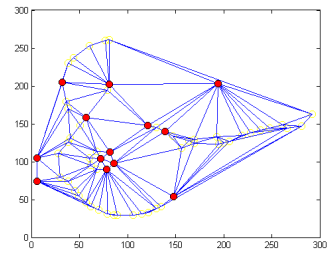
FIGURE 5.7: NODE SELECTION BY USING CLOSENESS CENTRALITY



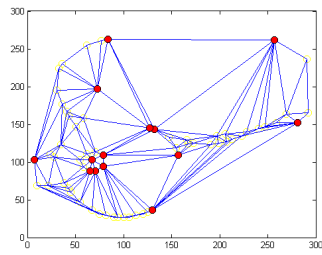
(a) Img1



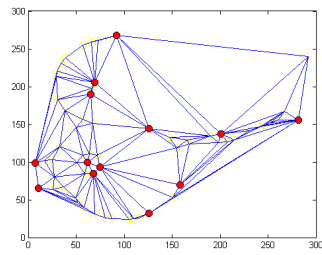
(b) Img2



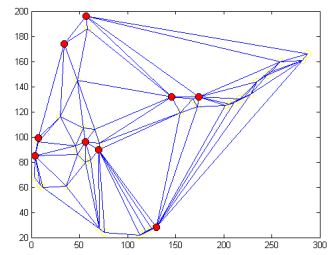
(c) Img3



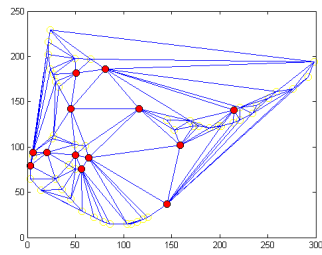
(d) Img4



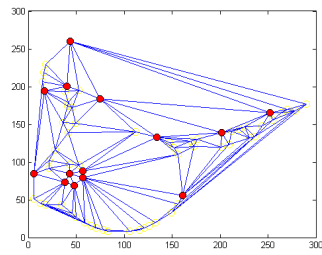
(e) Img5



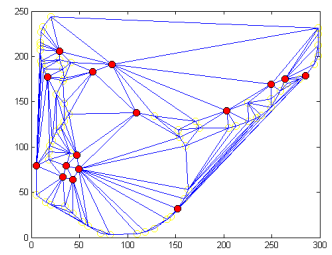
(f) Img6



(g) Img7

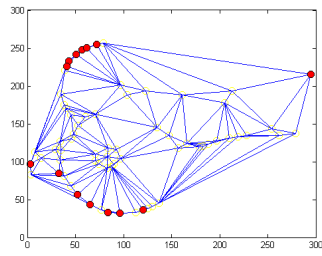


(h) Img8

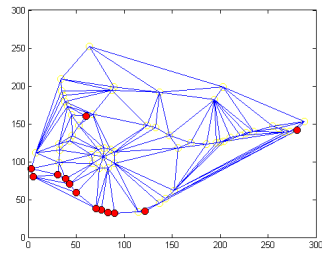


(i) Img9

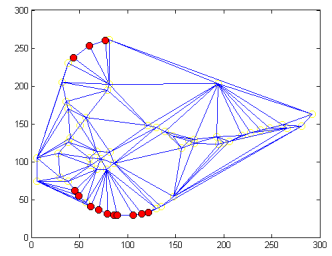
FIGURE 5.8: NODE SELECTION BY USING BETWEENNESS CENTRALITY



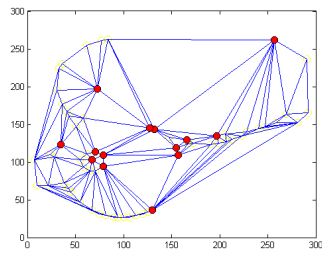
(a) Img1



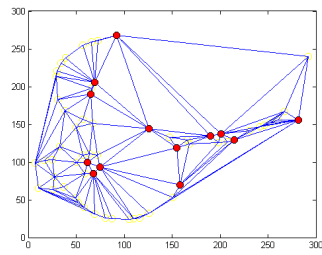
(b) Img2



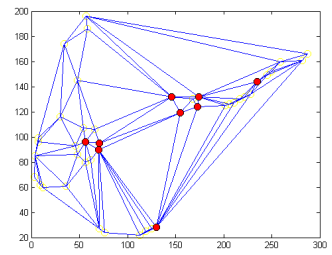
(c) Img3



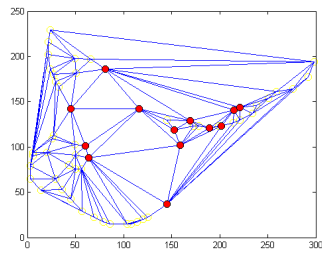
(d) Img4



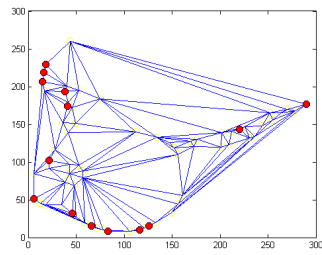
(e) Img5



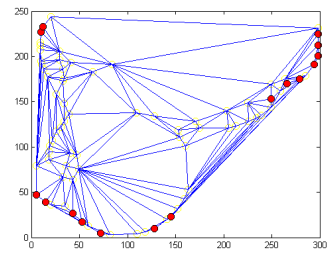
(f) Img6



(g) Img7

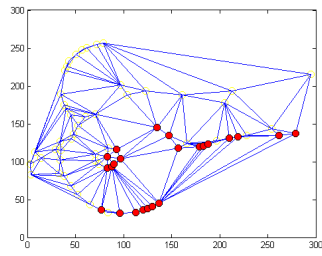


(h) Img8

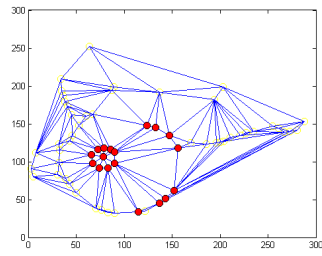


(i) Img9

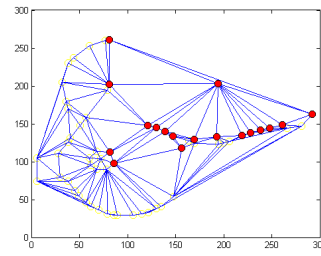
FIGURE 5.9: NODE SELECTION BY USING EIGENVECTOR CENTRALITY



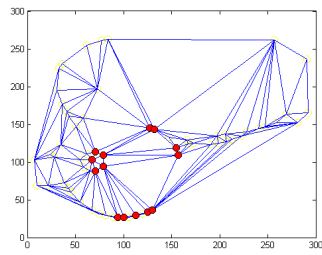
(a) Img1



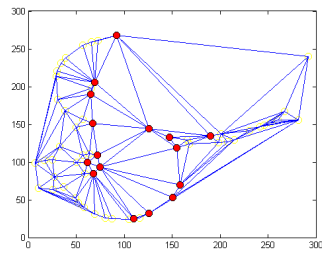
(b) Img2



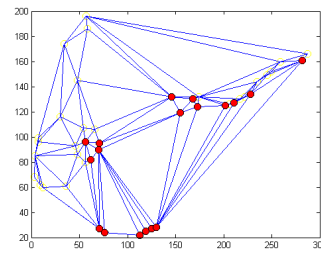
(c) Img3



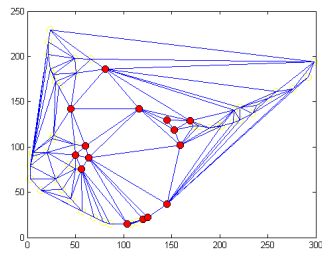
(d) Img4



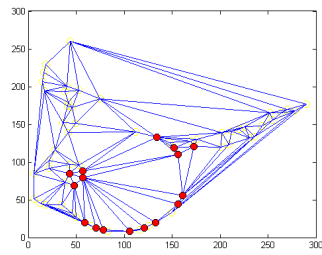
(e) Img5



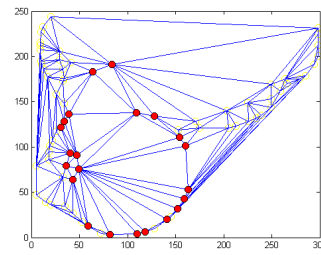
(f) Img6



(g) Img7



(h) Img8



(i) Img9

FIGURE 5.10: NODE SELECTION BY USING THE DOMINANT SET

## 5.3 Algorithm Description

The graph-based image matching algorithm can be described as:

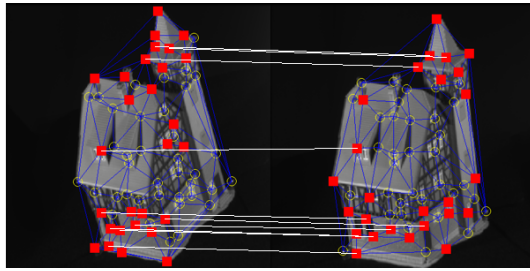
1. For given images, select the Delaunay graphs with the critical time  $t_c = 7$ .
2. Compute the heat kernel  $H_t$  at  $t_c = 7$  from an adjacency matrix or Laplacian matrix using Equation (2.42) or (2.38).
3. Select hot nodes using Definition (5.6).
4. Assign SIFT descriptors to the hot nodes.
5. Match the hot nodes by graph and SIFT matching algorithms.

## 5.4 Experimental Analysis

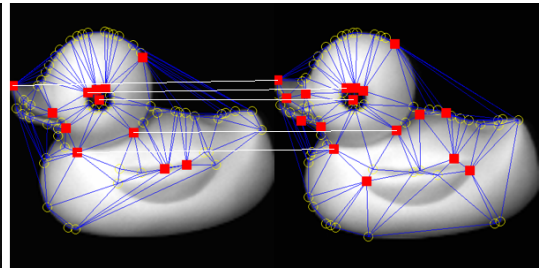
In this section, our method is tested on some benchmark images [16, 205, 206]. The images are taken at different views of objects against a black background. Illumination and scale are almost constant.

Firstly, some image matching examples are shown in Figure 5.11. The yellow circles denote graph nodes and the red squares denote hot nodes. The white lines denote the matches between images. As can be seen, new feature points are generated in subsequent images and some old feature points become undetectable because of the rotation of objects. This changes the graph structure slightly but visibly. However, the hot nodes are comparatively stable on a graph. As a result, our method is more accurate and faster than a full graph matching method.

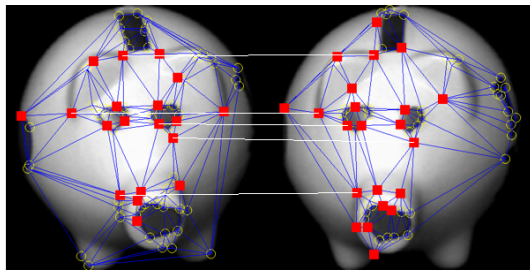
Secondly, the SIFT matching method [14] is applied to the same images shown in Figure 5.12. In Figure 5.12(a) and 5.12(f), only 10% of total matches are drawn for a better demonstration. The blue lines denote mismatches. In Figure 5.12(a), there are 3 mismatches. Figure 5.12(b) shows 10 matches including 3 meaningless background matches and 1 mismatch, whilst Figure 5.11(b) shows 5 correct matches and 0 mismatch by our method.



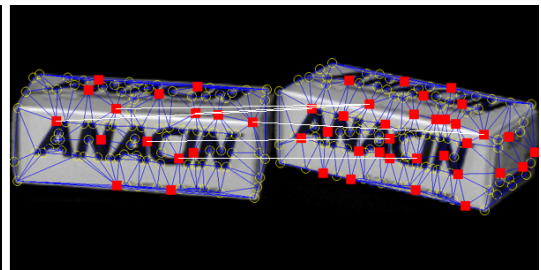
(a) CMU House



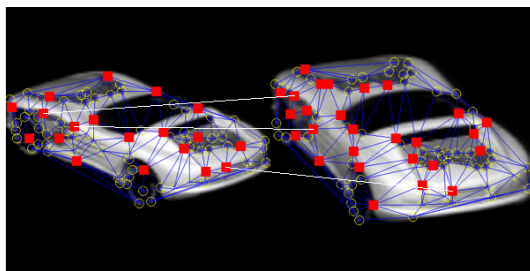
(b) COIL Duck



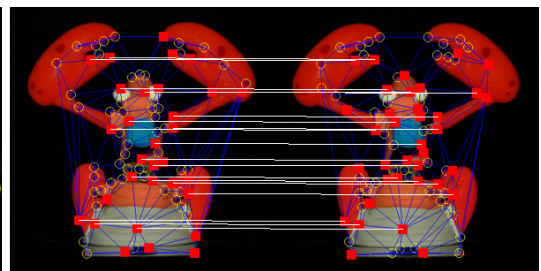
(c) COIL Pig



(d) COIL Anacin

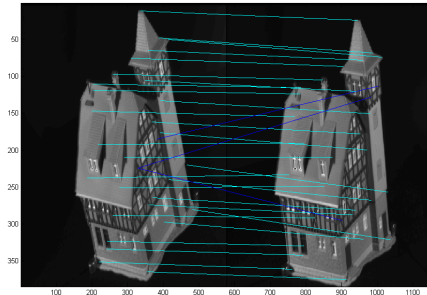


(e) COIL Car

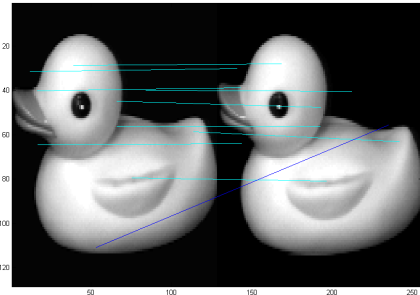


(f) Toy Lobster

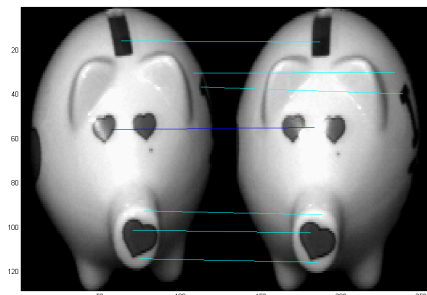
FIGURE 5.11: IMAGE MATCHING BY OUR METHOD



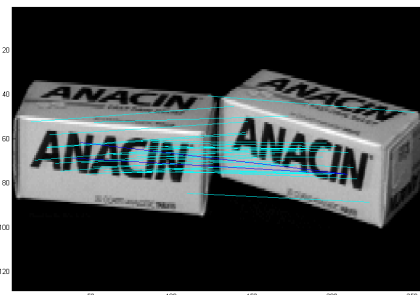
(a) CMU House



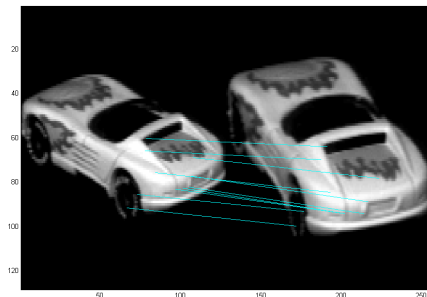
(b) COIL Duck



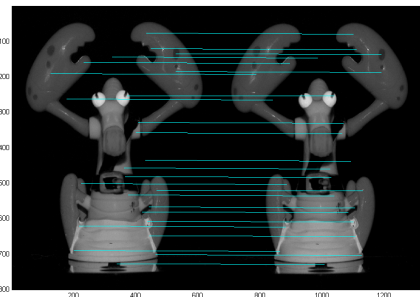
(c) COIL Pig



(d) COIL Anacin



(e) COIL Car



(f) Toy Lobster

FIGURE 5.12: IMAGE MATCHING BY SIFT





FIGURE 5.13: COIL-20 DATABASE (FROM [201])

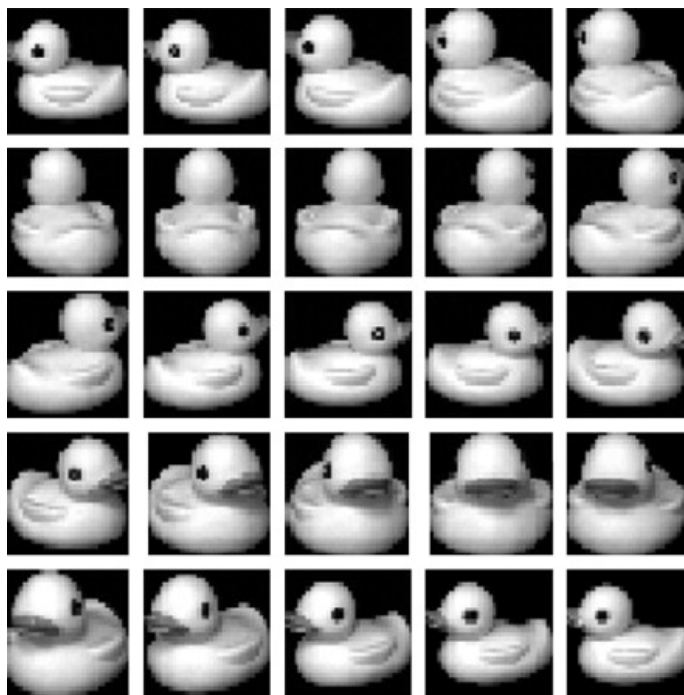
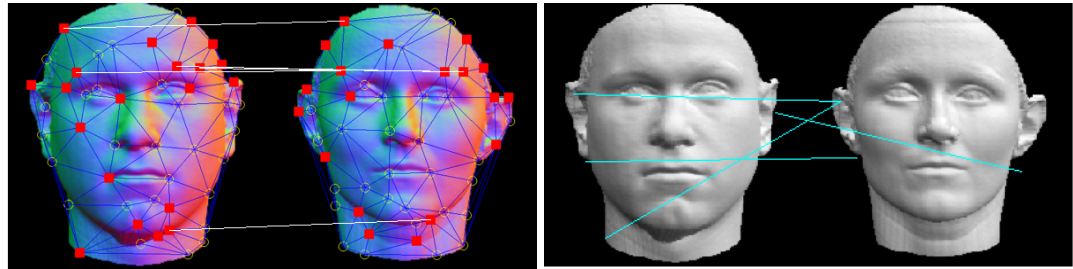
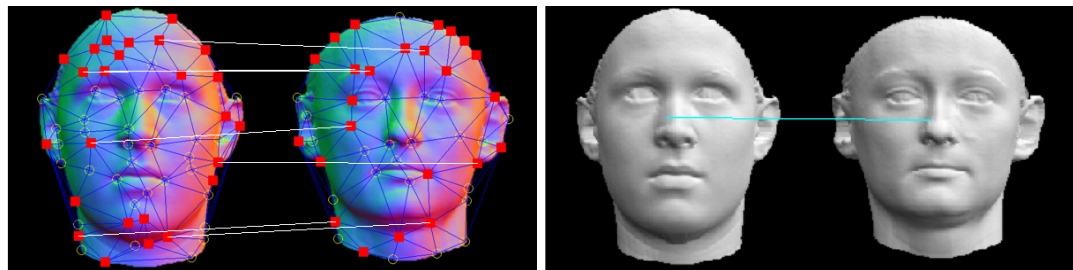


FIGURE 5.14: COIL DUCK IMAGES (FROM [201])



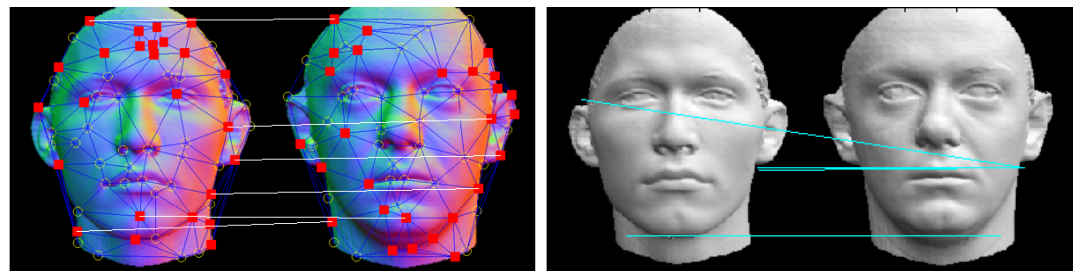
(a) Our method

(b) SIFT



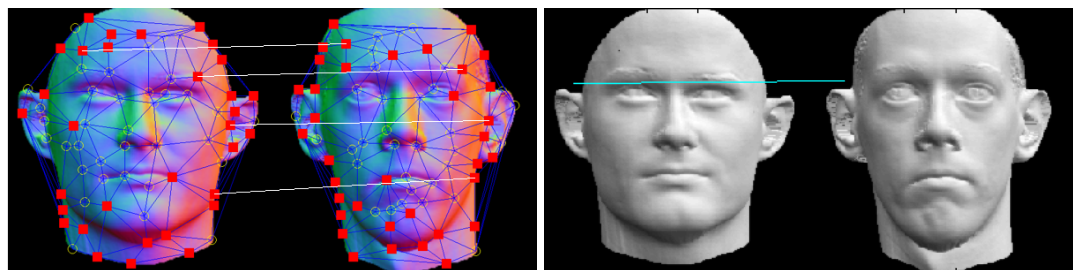
(c) Our method

(d) SIFT



(e) Our method

(f) SIFT



(g) Our method

(h) SIFT

FIGURE 5.15: FACE MATCHING COMPARISON

TABLE 5.2: IMAGE MATCHING AVERAGE ERROR COMPARISON

|            | Object number |     |     |     |     |     |     |     |   |     |
|------------|---------------|-----|-----|-----|-----|-----|-----|-----|---|-----|
|            | 1             | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9 | 10  |
| Our method | 0.8           | 1.3 | 0.4 | 0.3 | 0.2 | 0   | 0.6 | 0.1 | 0 | 0.2 |
| SIFT       | 2.1           | 1.4 | 0.8 | 0.5 | 0.5 | 0.3 | 0.5 | 0.3 | 0 | 0.4 |

|            | Object number |     |     |     |     |     |     |     |     |     |
|------------|---------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|            | 11            | 12  | 13  | 14  | 15  | 16  | 17  | 18  | 19  | 20  |
| Our method | 1.7           | 1.3 | 0.8 | 0.6 | 1.8 | 1.4 | 1.3 | 1.3 | 0.3 | 0.5 |
| SIFT       | 2.2           | 2   | 1.4 | 1.4 | 1.8 | 2   | 1.6 | 1.8 | 0.7 | 0.5 |

To quantitatively compare these two methods, experiments are conducted on COIL-20 images [201]. The objects of the full database and a set of duck images are shown in Figure 5.13 and 5.14. For each object, we randomly pick an image  $I_i$ , and then match it with one of its neighboring images in the range of  $I_{i-4}$  to  $I_{i+4}$ . This matching process is repeated 10 times for each object. The result is shown in Table 5.2. Clearly, the SIFT matching method generates more mismatches.

The Max Planck faces [200] are also used for comparison. The database contains images of 7 views of 200 laser-scanned heads. The heads are without hair and are further synthesized to not resemble any individuals. The result is shown in Figure 5.15. The faces matched in each row are from different persons. Obviously, our method has a better performance in detecting and matching same features from different faces. Especially in Figure 5.15(g) and 5.15(h), due to facial variances, there is only 1 background match by SIFT matching, in contrast to 4 correct matches by our method.

## 5.5 Conclusion

In this chapter, Delaunay graph characterization is studied. The diffusion process on a Delaunay graph is investigated and a node selection algorithm is proposed for graph-based image matching. The experimental analysis shows our method could select feature points and match images correctly and efficiently.

For future work, there are two directions. First, we want to extend our method to three-dimensional object matching. Second, we want to develop a graph-based motion tracking application.

# Chapter 6

## Conclusions and Future Work

In this chapter, our main contributions in data clustering and image matching are summarized. The strength and weakness of each method are discussed. The possible directions are also drawn for future work.

### 6.1 Contributions

#### 6.1.1 M1NN Agglomerative Clustering

In Chapter 3, our first contribution is the M1NN agglomerative clustering method. This method is compatible with different types of data and robust to noise. For normal agglomerative methods, the chaining phenomenon could cause 'over' clustering. Our parallel merging algorithm solves this problem effectively since outliers are excluded in merging steps.

Our second contribution is a cluster characterizing quantity derived from the path-based dissimilarity measure. The complexity of the path-based pairwise clustering algorithm was reckoned the same as that of an all-pairs-shortest-path problem. We proved it is actually much lower and equal to the complexity of an

MST problem. The cluster characterizing quantity is redefined from the bottleneck edge of the MST with its computational cost greatly reduced by our work.

Comparing with the state-of-the-art hierarchical clustering methods, clearly our method is more sophisticated and advanced. In our method, a new agglomerative clustering framework, rather than a similarity measure, is proposed. The elementary units of clusters are defined by CPs instead of single nodes. During the process of clustering, a parallel merging algorithm is applied to effectively prevent early merging of large clusters. Owing to these special features, our method has a better clustering performance and is extendable to many applications, e.g. data mining, image segmentation and manifold learning.

### **6.1.2 Modified Log-likelihood Clustering**

In Chapter 4, our first contribution is the MLL clustering expression based on the pioneering work of Robles-Kelly and Hancock. The new MLL clustering model, comparing with the ML criterion, contains additional information to measure and compare clusterings quantitatively and accurately.

Our second contribution is using the energy of a graph to measure the complexity of a clustering. The energy is computed by using the eigenvalues of the adjacency matrix of the clustering, hence the spectral property of the clustering is effectively added into the MLL model.

Apparently, our method is more convenient to use when clusterings need to be quantitatively compared without knowing the ground truth, i.e. the correct labels of the data points. Our method is also useful for selecting eigenvectors for optimized image segmentation or comparing graph partitions for the best cut.

### **6.1.3 Delaunay Graph Characterization and Graph-Based Image Matching**

In Chapter 5, our first contribution is a Delaunay graph characterization method based on diffusion process. The pairwise distances between graph nodes are estimated properly by normalized Euclidean distances derived from the heat kernel at the critical time.

Our second contribution is a graph-based image matching method. Hot nodes are defined on heat diffusion at the critical time. The comparison against other node selection methods shows that hot nodes are more stable and reliable for matching. SIFT descriptors are assigned to the hot nodes to develop a hybrid method for enhanced matching performance.

Essentially, our study contributes to a better understanding of Delaunay graph characterization. The critical time, as demonstrated, is a very interesting property of a Delaunay graph. Comparing with the commute time, which was considered as a fine approximation of the pairwise distance, our method provides a more accurate approximation computed from the heat kernel at the critical time. The critical time criterion is also useful for selecting the proper Delaunay graph representation for an image. In our graph-based image matching method, the critical time is used to define the hot nodes of a Delaunay graph, which are relatively stable across the graphs of a series of images and are suitable for establishing correspondences between images. As a result in the experiment, our method outperforms SIFT matching method by a lower error rate.

## **6.2 Limitations**

Generally our methods perform better than the state-of-the-art methods, but there are still some limitations to be mentioned.

In the M1NN clustering method, merging processes are controlled by the cluster characterizing quantity with a user-defined parameter. Once the parameter is set properly, merging errors can be greatly reduced and the impact of chaining phenomenon is minimized. However in most cases, clusters are generated more than expected. To solve this problem, a strong merging rule is derived from the classical agglomerative clustering algorithm to further process the clustering for the exact number of clusters. The experimental analysis demonstrates that the data points in an Euclidean space are grouped correctly by the strong merging rule. On the other hand, non-Euclidean data sets are not adequately supported. As a result, the clustering of such a data set is a little sparse although main clusters are located effectively. In the meantime, the input of cluster number is still required by our method, whilst there are many other methods attempting to automatically detect clusters without this number.

In the MLL clustering comparison method, since assumptions were made to simplify the probability and construct the Bernoulli model in the corresponding ML framework, our method may not work satisfactorily in practice when the assumptions cannot hold true. And because the assumptions were made for hard clustering problems, it is difficult to extend our method for soft clustering. Moreover, our method is limited by the condition that it can only compare the clusterings with the same number of clusters, which seems less practical sometimes. Also, our method may have slightly varied performance for the same data in different graph representations.

In the graph-based image matching method, a common critical time is set to select Delaunay graphs for matching a sequence of images. As demonstrated, all the graphs with the same and proper critical time are of similar size and optimally represent the images. However the appropriate critical time is currently decided heuristically without being justified. Furthermore, the critical time used



for hot node selection is also decided by experiments. Thus the performance of our method is not theoretically guaranteed. Comparing with the SIFT matching method, our method has a lower error rate. However because SIFT descriptors are used by our method to provide additional information for the graph nodes which are extracted from an image by the Harris corner detector, the number of matches is sometimes much lower too.

### 6.3 Future Work

Apparently there are a lot of work to do to improve our methods for the limitations addressed in last section.

For the M1NN clustering method, the strong merging rule needs to be investigated for the data sets in non-Euclidean spaces. The problem may be caused by the distance measure which is defined differently from the Euclidean distance, then the structure of a cluster cannot be fully captured. As the cluster number is currently a terminating condition for the CQ merging algorithm, it will be necessary to replace this condition so that our clustering method can work automatically without this number. Our method should also be optimized for data mining and image segmentation.

For the MLL clustering comparison method, the assumptions should be further inspected and changed for overlapped clusters and relaxed values of cluster memberships. Research should be conducted to modify the MLL model for the clusterings with different number of clusters. Other definitions of graph energy could be used so that the performance may become more stable and less sensitive to the graph representation. Our method could also be adapted to develop a feature selection algorithm in future.

For the graph-based image matching method, the critical time still needs more

research from the aspects of graph theory and complex networks. In the meantime, different feature detectors and descriptors can be adopted to improve the matching accuracy and increase the number of matches. It is also feasible to extend our method for three-dimensional object matching since the surface of an object is usually meshed with triangles that our current research can be migrated smoothly. A graph-based motion tracking application could be developed based on our method too.

# Glossary of Notation

|                 |  |
|-----------------|--|
| $G(V, E)$       | Graph with vertex set $V$ and edge set $E$ |
| $A$             | Adjacency matrix                           |
| $\mathcal{A}$   | Normalized adjacency matrix                |
| $\text{deg}(u)$ | Degree of the vertex $u$                   |
| $D$             | Degree matrix or data, see context         |
| $L$             | Laplacian matrix                           |
| $\mathcal{L}$   | Normalized Laplacian matrix                |
| $I$             | Identity matrix or image, see context      |
| $\Phi$          | Eigenvector matrix                         |
| $\Lambda$       | Eigenvalue matrix                          |
| $\phi_i$        | The $i$ -th smallest eigenvector           |
| $\lambda_i$     | The $i$ -th smallest eigenvalue            |
| $\Omega$        | Set of clusters                            |
| $\mathcal{H}$   | Hypothesis                                 |
| $L_c$           | Coding length                              |
| $P$             | Probability                                |
| $H_t$           | Heat kernel                                |
| $Tr$            | Trace operator                             |
| $\det$          | Determinant operator                       |

|                      |  |
|----------------------|--|
| CP                   | Couple, the two mutually nearest points  |
| $\kappa$             | Cluster characterizing quantity          |
| $\zeta$              | Zeta function                            |
| $\Gamma$             | Gamma function                           |
| $\Delta$             | Laplace operator                         |
| $G$                  | Green's function                         |
| $\mathcal{G}$        | Normalized Green's function              |
| $Q$                  | Hitting time                             |
| $CT$                 | Commute time                             |
| $k$                  | Parameter, see context for exact meaning |
| $s_{iw}$             | Cluster membership indicator             |
| $S$                  | Clustering matrix to denote a clustering |
| $\Theta$             | Clustering space                         |
| $E_{s_w}$            | Graph energy of the cluster $w$          |
| $\beta$              | Inverse temperature                      |
| $\mathcal{DL}(A, S)$ | Description length function              |
| $\mathcal{L}(A, S)$  | Log-likelihood function                  |
| $d_G$                | Geodesic distance                        |
| $d_{EN}$             | Normalized Euclidean distance            |
| $n_h$                | Hot node                                 |

## References

- [1] D. A. M. Noiton and P. A. Alspach. Founding clones, inbreeding, coancestry, and status number of modern apple cultivars. *Journal of the American Society for Horticultural Science*, 121(5):773-782, 1996.
- [2] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. PAMI*, 22(8):888-905, 2000.
- [3] M. Iwayama and T. Tokunaga. Cluster-based text categorization: A comparison of category search strategies. *SIGIR '95 Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.273-280, 1995.
- [4] P. Baldi and G. W. Hatfield. DNA microarrays and gene expression. *Cambridge University Press*, 2002.
- [5] S. Basu, I. David and K. Wagstaff. Constrained clustering: Advances in algorithms, theory and applications. *Chapman and Hall/CRC*, 2008.
- [6] R. Duda, P. Hart and D. Stock. Pattern Classification. *New York: John Wiley & Sons*, 2001.
- [7] J. He, A. H. Tan, S. Y. Sung and C. L. Tan. On quantitative evaluation of clustering systems. *Information Retrieval and Clustering*, 2002.

- [8] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651-666, 2010.
- [9] U. von Luxburg, M. Belkin and O. Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, 36(2):555-586, 2008.
- [10] S. Guattery and G. L. Miller. On the quality of spectral separators. *SIAM Journal on Matrix Analysis and Applications*, pp.701-719, 1998.
- [11] T. Xiang and S. Gong. Spectral clustering with eigenvector selection. *Pattern Recognition*, 41(3):1012-1029, 2008.
- [12] R. Haralick. Ridges and valleys on digital images. *Computer Vision, Graphics and Image Processing*, 22(1):28-38, 1983.
- [13] B. Luo and E. R. Hancock. Iterative procrustes alignment and the EM algorithm. *Image Vision Computing*, 20(5-6):377-396, 2002.
- [14] D. G. Lowe. Object recognition from local scale-invariant features. *Proceedings of the International Conference on Computer Vision*, Vol.2, p.1150, 1999
- [15] H. Qiu and E. R. Hancock. Graph matching and clustering using spectral partitions. *Pattern Recognition*, 39(1):22-34, 2006.
- [16] P. Ren. Structural learning based on Ihara coefficients and hypergraphs. *PhD Thesis, University of York*, 2010.
- [17] R. J. Sternberg. Cognitive psychology. *Thomson Wadsworth*, 2003.
- [18] N. Ahuja and M. Tuceryan. Extraction of early perceptual structure in dot patterns: Integrating region, boundary, and component gestalt. *CVGIP*, 48(3):304-356, 1989.

- [19] J. Han and M. Kamber. Data mining: Concepts and techniques. *Morgan Kaufmann*, 2006.
- [20] C. R. Lin and M. S. Chen. Combining partitional and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):145-159, 2005.
- [21] P. Tan, M. Steinbach and V. Kumar. Introduction to data mining. *Addison-Wesley Longman*, 2005.
- [22] E. F. Krause. Taxicab geometry. *Dover*, 1987.
- [23] P. C. Mahalanobis. On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49-55, 1936.
- [24] R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 26(2):147-160, 1950.
- [25] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707-710, 1966.
- [26] G. J. Székely and M. L. Rizzo. Hierarchical clustering via joint between-within distances: Extending Ward's minimum variance method. *Journal of Classification*, 22(2):151-183, 2005.
- [27] R. Sibson. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal (British Computer Society)*, 16(1):30-34, 1973.
- [28] T. Sorensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. *Biologiske Skrifter*, Vol.5, pp.1-34, 1948.

- [29] P. H. A. Sneath and R. R. Sokal. Numerical taxonomy: The principles and practice of numerical classification. *San Francisco: Freeman*, 1973.
- [30] A. K. Jain, M. N. Murty and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264-323, 1999.
- [31] T. Zhang, R. Ramakrishnan, and M. Linvy. BIRCH: An efficient data clustering method for very large data sets. *Proceedings of the ACM SIGMOD Conference*, pp.103-114, 1996.
- [32] S. Guha, R. Rastogi and K. Shim. CURE: An efficient clustering algorithm for large data sets. *Proceedings of the ACM SIGMOD Conference*, pp.73-84, 1998.
- [33] G. Karypis, E. H. Han and V. Kumar. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer*, 32(8):68-75, 1999.
- [34] V. Gaede and O. Gunther. Multidimensional access methods. *ACM Computing Surveys*, 30(2):170-231, 1998.
- [35] P. Berkhin. A survey of clustering data mining techniques. *Grouping Multidimensional Data*, pp.25-71, 2006.
- [36] Y. Ma, H. Derksen, W. Hong and J. Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *IEEE Trans. PAMI*, 29(9):1546-1562, 2007.
- [37] A. Y. Yang, J. Wright, Y. Ma and S. S. Sastry. Unsupervised segmentation of natural images via lossy data compression. *Computer Vision and Image Understanding*, 110(2):212-225, 2008.
- [38] D. Zhao and X. Tang. Cyclizing clusters via Zeta function of a graph. *NIPS*, pp.1953-1960, 2008.



- [39] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, Vol.1, pp.281-297, 1967.
- [40] G. P. Babu and M. N. Murty. Simulated annealing for selecting optimal initial seeds in the k-means algorithm. *Indian J. pure appl. Math*, 25(1-2):85-94, 1994.
- [41] A. M. Fahim, G. Saake, A. M. Salem, F. A. Torkey and M. A. Ramadan. K-means for spherical clusters with large variance in sizes. *Engineering and Technology*, pp.177-182, 2008.
- [42] M. Sato, Y. Sato and L. C. Jain. Fuzzy clustering models and applications. *Physica-Verlag*, 1997.
- [43] G. J. McLachlan and T. Krishnan. The EM algorithm and extensions. *John Wiley & Sons*, 1997.
- [44] J. Y. Choi. Unsupervised learning of finite mixture models with deterministic annealing for large-scale data analysis. *PhD thesis, Indiana University*, 2012.
- [45] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *J. Science*, Vol.315, pp.972-977, 2007.
- [46] X. Zhang. Contributions to large scale data clustering and streaming with affinity propagation. Application to autonomic grids. *PhD thesis, Université Paris-Sud*, 2010.
- [47] X. Zhang, C. Furtlehner and M. Sebag. Data streaming with affinity propagation. *In European Conference on Machine Learning and Practice of Knowledge Discovery in Databases*, pp.628-643, 2008.

- [48] X. Zhang, M. Sebag and C. Germain-Renaud. Multi-scale real-time grid monitoring with job stream mining. *In 9th IEEE International Symposium on Cluster Computing and the Grid*, pp.420-427, 2009.
- [49] J. Han, M. Kamber and A. K. H. Tung. Spatial clustering methods in data mining: A survey. *Geographic Data Mining and Knowledge Discovery*, 2001.
- [50] M. Ester, H. Kriegel, J. Sander and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp.226-231, 1996.
- [51] J. Sander, M. Ester, H. Kriegel and X. Xu. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169-194, 1998
- [52] X. Xu, M. Ester, H. Kriegel, and J. O. Sander. A nonparametric clustering algorithm for knowledge discovery in large spatial data sets. *Proc. IEEE Int. Conf. on Data Engineering*, 1998.
- [53] M. Ankerst, M. M. Breunig, H. Kriegel and J. O. Sander. OPTICS: Ordering points to identify the clustering structure. *ACM SIGMOD international conference on Management of data*, pp.49-60, 1999.
- [54] E. Achtert, C. Böhm, H. Kriegel, P. Kröger, I. Müller-Gorman and A. Zimek. Finding hierarchies of subspace clusters. *LNCS: Knowledge Discovery in Databases*, 2006.
- [55] C. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. On Computer*, 20(1):68-86, 1971.

- [56] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298-305, 1973.
- [57] M. Meila and J. Shi. A random walks view of spectral segmentation. *Conference in AI and Statistics*, 2001.
- [58] S. Yu and J. Shi. Multiclass spectral clustering. *International Conference on Computer Vision*, Vol.2, p.313, 2001.
- [59] L. Grady and E. L. Schwartz. Isoperimetric graph partitioning for image segmentation. *IEEE Trans. PAMI*, 28(3):469-475, 2006.
- [60] W. N. Anderson and T. D. Morley. Eigenvalues of the Laplacian of a graph. *Linear and Multilinear Algebra*, 1985.
- [61] M. Pavan and M. Pelillo. Dominant sets and pairwise clustering. *IEEE Trans. PAMI*, 29(1):167-172, 2007.
- [62] J. W. Weibull. Evolutionary game theory. *MIT Press*, 1995.
- [63] M. Pavan and M. Pelillo. Dominant sets and hierarhical clustering. *Proc. IEEE Intl Conf. Computer Vision*, Vol.2, p.362, 2003.
- [64] J. H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236-244, 1963.
- [65] K. A. Heller and Z. Ghahramani. Bayesian hierarchical clustering. *ICML '05 Proceedings of the 22nd international conference on Machine learning*, pp.297-304, 2005.
- [66] Dendrogram.  
<http://www.mathworks.co.uk/help/toolbox/stats/dendrogram.html>

- [67] P. Drineas, A. Frieze, R. Kannan, S. Vempala and V. Vinay. Clustering large graphs via the singular value decomposition. *Machine learning*, 56(1-3):9-33, 1999.
- [68] D. C. MacKay. Information theory, inference and learning algorithms. *Cambridge University Press*, 2003.
- [69] J. M. Peña, J. A. Lozano and P. Larrañaga. An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recognition Letters*, 20(10):1027-1040, 1999.
- [70] L. Kaufmann and P. J. Rousseeuw. Finding groups in data: An introduction to cluster analysis. *John Wiley*, 1990.
- [71] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Trans. PAMI*, 15(11):1101-1113, 1993.
- [72] C. H. Papadimitriou. Computational complexity. *Addison-Wesley*, 1994.
- [73] G. H. Golub and C. F. Van Loan. Matrix computations. *John Hopkins Press*, 1989.
- [74] M. Fiedler. Laplacian of graphs and algebraic connectivity. *Combinatorics and Graph Theory*, Vol.25, pp.57-70, 1989.
- [75] J. Cheeger. A lower bound for the smallest eigenvalue of the Laplacian. *Princeton University Press*, 1970.
- [76] B. Mohar. Isoperimetric numbers of graphs. *J. Combin. Theory Ser. B*, 47(3):274-291, 1989.

- [77] J. Arfken and H. J. Weber. *Mathematical methods for physicists*. *J. Academic Press, New York*, 1985.
- [78] B. J. Frey and D. Dueck. Mixture modeling by affinity propagation. *J. NIPS*, pp.379-386, 2005.
- [79] B. J. Frey. Affinity propagation: New algorithms, results and applications. *J. International Workshop on Cognitive Dynamic Systems and Their Application*, 2008.
- [80] V. K. Balakrishnan. *Graph Theory*. *McGraw-Hill*, 1997.
- [81] N. L. Biggs, E. K. Lloyd and R. J. Wilson. *Graph Theory, 1736-1936*. *Oxford University Press*, 1986.
- [82] D. Zwillinger. *CRC Standard Mathematical Tables and Formulae*. *Chapman & Hall/CRC*, 2002.
- [83] C. Chen, W. K. Härdle and A. Unwin. *Handbook of data visualization*. *Springer Handbooks of Computational Statistics*, 2008.
- [84] F. Buckley and F. Harary. *Distance in graphs*. *Redwood City, CA: Addison-Wesley*, 1990.
- [85] B. Delaunay. Sur la sphère vide. *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvoennykh Nauk*, pp.793-800, 1934.
- [86] M. Abellanas, P. Bose, J. García, F. Hurtado, C. M. Nicolás and P. Ramos. On structural and graph theoretic properties of higher order Delaunay graphs. *Int. J. Comput. Geometry Appl.*, 19(6):595-615, 2009.
- [87] A. Okabe, B. Boots, K. Sugihara and S. N. Chiu. *Spatial tessellations C concepts and applications of Voronoi diagrams*. *John Wiley*, 2000.

- [88] Delaunay graph and voronoi tessellation.  
<http://www.wikipedia.org/wiki/Delaunaycircumcircles>  
<http://www.wikipedia.org/wiki/DelaunayVoronoi>
- [89] R. Diestel. Graph theory. *Berlin, New York: Springer-Verlag*, 2005.
- [90] R. G. Gallager, P. A. Humblet and P. M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Transactions on Programming Languages and Systems*, 5(1):66-77, 1983.
- [91] Minimum spanning tree.  
<http://www.wikipedia.org/wiki/Minimumspanningtree>
- [92] D. M. Cvetkovic and H. Sachs. Spectra of graphs: Theory and applications. *New York: Wiley*, 1998.
- [93] Laplacian matrix.  
<http://en.wikipedia.org/wiki/Laplacianmatrix>.
- [94] P. McCullagh. Marginal likelihood for distance matrices. *Statistica Sinica*, 2009.
- [95] J. Aldrich. Eigenvalue, eigenfunction, eigenvector, and related terms. *Earliest Known Uses of Some of the Words of Mathematics*, 2006.
- [96] F. R. K. Chung. Spectral graph theory. *CBMS Regional Conference Series in Mathematics*, 1996.
- [97] N. L. Biggs. Algebraic graph theory. *Cambridge, England: Cambridge University Press*, 1993.

- [98] A. Pothen, H. D. Simon and K. P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Analytical Applications*, 11(3):430-452, 1990.
- [99] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465-471, 1978.
- [100] J. Rissanen. Universal coding, information, prediction and estimation. *IEEE Transactions on Information Theory*, 30(4):629-636, 1984.
- [101] J. Rissanen, T. P. Speed and B. Yu. Density estimation by stochastic complexity. *IEEE Transactions on Information Theory*, 38(2):315-323, 1992.
- [102] J. Rissanen. MDL denoising. *IEEE Transactions on Information Theory*, 46(7):2537-2543, 2000.
- [103] A. Torsello and E. Hancock. Learning mixtures of tree-unions by minimizing description length. *EMMVCVPR'03*, pp.130-146, 2003.
- [104] P. Grünwald. The minimum description length principle. *MIT Press*, 2007.
- [105] P. Grünwald et al. Advances in minimum description length. *MIT Press*, 2005.
- [106] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 1948.
- [107] R. M. Fano. The transmission of information. *Technical Report No. 65 USA: Research Laboratory of Electronics at MIT*, 1949.
- [108] B. Luo, R. C. Wilson and E. R. Hancock. Spectral embedding of graphs. *Pattern Recognition*, 36(10):2213-2230, 2003.

- [109] K. Riesen and H. Bunke. Approximate graph edit distance computation by means of bipartite graph matching. *Image and Vision Computing*, 27(7):950-959, 2009.
- [110] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Trans. PAMI*, 10(5):695-703, 1988.
- [111] L. Shapiro and J. Brady. Feature-based correspondence: an eigenvector approach. *Image Vision Computing*, 10(5):283-288, 1992.
- [112] M. Juvan and B. Mohar. Optimal linear labelings and eigenvalues of graphs. *Disc. Appl. Math*, 36(2):153-168, 1992.
- [113] J. E. Atkins, E. G. Boman and B. Hendrickson. A spectral algorithm for seriation and the consecutive ones problem. *SIAM Journal on Computing*, 28(1):297-310, 1998.
- [114] A. Shokoufandeh, S. J. Dickinson, K. Siddiqi and S. W. Zucker. Indexing using a spectral encoding of topological structure. *In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp.2491-2497, 1999.
- [115] A. Robles-Kelly and E. R. Hancock. Graph edit distance from spectral seriation. *IEEE Trans. PAMI*, 27(3):365-378, 2005.
- [116] A. Robles-Kelly and E. R. Hancock. String edit distance, random walks and graph matching. *IJPRAI*, 18(3):315-327, 2004.
- [117] B. Luo, R. C. Wilson and E. R. Hancock. A spectral approach to learning structural variations in graphs. *Pattern Recognition*, 39(6):1188-1198, 2006.
- [118] H. Qiu and E. R. Hancock. Commute times, discrete Green's functions and graph matching. *13th International Conference on Image Analysis and Processing*, pp.454-462, 2005.



- [119] H. Qiu and E. R. Hancock. Commute times for graph spectral clustering. *CAIP*, pp.128-136, 2005.
- [120] X. Bai, R. C. Wilson and E. R. Hancock. Characterising graphs using the heat kernel. *BMVC*, 2005
- [121] Xiao Bai, E. R. Hancock and R. C. Wilson. Graph characteristics from the heat kernel trace. *Pattern Recognition*, 42(11):2589-2606, 2009.
- [122] P. Flajolet, X. Gourdon and P. Dumas. Mellin transforms and asymptotics: Harmonic sums. *Theoretical Computer Science*, 144(1-2):3-58, 1995.
- [123] F. R. K. Chung and S. T. Yau. Discrete green's functions. *In J. Combinatorial Theory*, 91(1-2):191-214, 2000.
- [124] H. Qiu and E. R. Hancock. Spanning trees from the commute times of random walks on graphs. *ICIAR*, pp.375-385, 2006.
- [125] J. Nieminen. On the centrality in a graph. *Scandinavian Journal of Psychology*, 15(1):332-336, 1974.
- [126] L. C. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks* 1, 1(3):215-239, 1979.
- [127] S. Skiena. Implementing discrete mathematics: Combinatorics and graph theory with Mathematica. *Reading, MA:Addison-Wesley*, pp. 225-253, 1990.
- [128] G. Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581-603, 1966.
- [129] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35-41, 1977.

- [130] P. Bonacich. Power and centrality: A family of measures. *The American Journal of Sociology*, 92(5):1170-1182, 1987.
- [131] S. Brin and L. Page. The PageRank citation ranking: Bringing order to the web. *Stanford InfoLab*, 1998.
- [132] J. P. Keener. The Perron-Frobenius Theorem and the ranking of football teams. *SIAM Review*, 35(1):80-93, 1993.
- [133] J. Canny. A computational approach to edge detection. *IEEE Trans. PAMI*, 8(6):679-698, 1986.
- [134] J. Shi and C. Tomasi. Good features to track. *9th IEEE Conference on Computer Vision and Pattern Recognition*, pp.593-600, 1994.
- [135] T. Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *International Journal of Computer Vision*, 11(3):283-318, 1993.
- [136] H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. *Tech Report CMU-RI-TR-3 Carnegie-Mellon University*, 1980.
- [137] C. G. Harris and M. J. Stephens. A combined corner and edge detector. *In Proceedings of Fourth Alvey Vision Conference*, pp.147-151, 1988.
- [138] R. Szeliski. Computer vision: Algorithms and applications. *Springer*, 2011.
- [139] W. Feller. An introduction to probability theory and its applications. *Wiley*, 1971.
- [140] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. *In Proceedings of the 8th International Conference on Computer Vision*, pp.128-142, 2002.

- [141] J. A. Noble. Descriptions of image surfaces. *PhD thesis, Dept. of Engineering Science, Oxford University*, 1989
- [142] M. Brown, R. Szeliski and S. Winder. Multi-image matching using multi-scale oriented patches. *CVPR*, Vol.1, pp.510-517, 2005
- [143] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91-110, 2004.
- [144] S. Edelman, N. Intrator and T. Poggio. Complex cells and object recognition. *Unpublished manuscript: <http://kybele.psych.cornell.edu/edelman/archive.html>*, 1997.
- [145] J. G. Auguston and J. Minker. An analysis of some graph theoretical clustering techniques. *J. ACM*, 17(4):571-588, 1970.
- [146] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Trans. Neural Networks*, 16(3):645-678, 2005.
- [147] M. Pelillo. What is a clustering? Perspectives from game theory. *NIPS Workshop*, 2009.
- [148] K. C. Gowda and G. Krishna. Agglomerative clustering using the concept of mutual nearest neighbourhood. *PR*, 10(2):105-112, 1978.
- [149] J. Francos, H. Permuter and I. H. Jermyn. Gaussian mixture models of texture and colour for image database retrieval. *ICASSP*, pp.25-88, 2003.
- [150] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.*, 7(1):48-50, 1956.
- [151] J. C. Gower and G. J. S. Ross. Minimum spanning trees and single linkage cluster analysis. *Journal of the Royal Statistical Society*, 18(1):54-64, 1969.

- [152] F. J. Rohlf. Single-link clustering algorithms. *Handbook of Statistics*, 1982.
- [153] B. Fischer and J. M. Buhmann. Path-based clustering for grouping of smooth curves and texture segmentation. *IEEE Trans. PAMI*, 25(4):513-518, 2003.
- [154] B. Fischer, T. Zöllner, and J. M. Buhmann. Path based pairwise data clustering with application to texture segmentation. *EMMVCVPR*, pp.235-250, 2001.
- [155] S. Pettie. On the shortest path and minimum spanning tree problems. *Doctoral Dissertation, The University of Texas at Austin*, 2003.
- [156] A. K. Nemani and R. K. Ahuja. Minimum spanning trees. *Wiley Encyclopedia of Operations Research and Management Science*, 2011.
- [157] H. Wang and E. Hancock. Probabilistic relaxation labelling using the fokker-planck equation. *Pattern Recognition*, 41(11):3393-3411, 2008.
- [158] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846-850, 1971.
- [159] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193-218, 1985.
- [160] A. L. N. Fred and J. M. N. Leitão. A new cluster isolation criterion based on dissimilarity increments. *IEEE Trans. on PAMI*, 25(8):944-958, 2003.
- [161] C. B. D. J. Newman, S. Hettich and C. Merz. UCI repository of machine learning databases, 1998.
- [162] I. Fischer and J. Poland. New methods for spectral clustering. *Technical Report IDSIA-12-04*, 2004.

- [163] S. Sarkar and K. L. Boyer. Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. *Computer Vision Image Understanding*, 71(1):110-136, 1998.
- [164] A. Torsello. Matching hierarchical structures for shape recognition. *PhD Thesis, University of York*, 2004.
- [165] A. Robles-Kelly and E. R. Hancock. A maximum likelihood framework for iterative eigendecomposition. *In Proc. Int. Conf. Computer Vision*, pp.654-661, 2001.
- [166] V. Silva, J. B. Tenenbaum and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319-2323, 2000.
- [167] A. Robles-Kelly and E. Hancock. A probabilistic spectral framework for grouping and segmentation. *Pattern Recognition*, 37(7):1387-1405, 2004.
- [168] E. Sober. Let's razor Occam's razor. *Explanation and Its Limits, Cambridge University Press*, 1994.
- [169] J. Rissanen. Information and complexity in statistical modeling. *Springer*, 2007.
- [170] C. Kittel and H. Kroemer. Thermal physics. *United States of America: W. H. Freeman and Company*, 1980.
- [171] J. Day and W. So. Graph energy change due to edge deletion. *Linear Algebra and its Applications*, 428(8-9):2070-2078, 2008.
- [172] P. McCullagh and J. Nelder. Generalized linear models. *Boca Raton: Chapman and Hall/CRC*, 1989.

- [173] I. Gutman. The energy of a graph. *Ber. Math. Stat. Sect. Forschungszentrum Graz.*, Vol.103, pp.1-22, 1978.
- [174] B. J. McClelland. Properties of the latent roots of a matrix: The estimation of  $\pi$ -electron energies. *J. Chem. Phys.*, Vol.54, pp.640-643, 1971.
- [175] I. Gutman. The energy of a graph: Old and new results. *A. Betten, A. Kohnert, R. Laue, A. Wassermann (Eds.), Algebraic Combinatorics and Applications, Springer-Verlag, Berlin*, pp.196C211, 2001.
- [176] R. Balakrishnan. The energy of a graph. *Linear Algebra and its Applications*, 387:287-295, 2004.
- [177] M. Kamal Kumar. Relation between domination number, energy of graph and rank. *International Journal of Mathematics and Scientific Computing*, 1(1):58-61, 2011.
- [178] I. Gutman and B. Zhou. Laplacian energy of a graph. *Linear Algebra and its Applications*, 44:29-37, 2006.
- [179] B. Xiao, Y. Z. Song and P. Hall. Learning invariant structure for object identification by using graph methods. *Computer Vision and Image Understanding*, 115(7):1023-1031, 2011.
- [180] P. Erdős and A. Rényi. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5:17-61, 1960.
- [181] C. S. Chekuri, A. V. Goldberg, D. R. Karger, M. S. Levine and C. Stein. Experimental study of minimum cut algorithms. *Proc. 8th Annual ACM-SIAM Symp. on Discrete Algorithms*, pp.324-333, 1997.

- [182] J. Sun, M. Ovsjanikov and L. Guibas. A concise and provably informative multi-scale signature-based on heat diffusion. *Computer Graphics Forum*, pp.1383-1392, 2009.
- [183] E. Estrada and J. A. Rodriguez-Velazquez. Subgraph centrality in complex networks. *Phys.Lett* 319, Vol.71, No.5, 2005.
- [184] E. Estrada and N. Hatano. Statistical-mechanical approach to subgraph centrality in complex networks. *Chemical Physics Letter*, pp. 247-251, 2009.
- [185] D. J. Higham, E. Estrada and N. Hatano. Communicability and multipartite structure in complex networks at negative absolute temperatures. *Physical Review*, Vol.78, No.2, 2008.
- [186] F. Escolano, E. R. Hancock and M. A. Lozano. Heat diffusion: Thermodynamic depth complexity of networks. *Physical Review*, Vol.85, No.3, 2012.
- [187] F. A. Haight. Handbook of the Poisson distribution. *New York: John Wiley & Sons*, 1967.
- [188] M. Volt. Asymptotics of heat kernels on projective spaces of large dimensions and on disk hypergroups. *Math. Nachr.*, pp.225-238, 1998.
- [189] L. P. Chew. There is a planar graph almost as good as the complete graph. *Proceedings of the Second Symposium on Computational Geometry*, pp.169-177, 1986.
- [190] D. P. Dobkin, S. J. Friedman and K. J. Supowit. Delaunay graphs are almost as good as complete graphs. *Journal Discrete & Computational Geometry*, pp.20-26, 1990.
- [191] J. M. Keil and C. A. Gutwin. Classes of graphs which approximate the complete Euclidean graph. *Discrete and Computational Geometry*, 7(1):13-28, 1992.

- [192] G. Narasimhan and M. Smid. Geometric spanner networks. *Cambridge University Press*, 2007.
- [193] X. Bai. Heat kernel analysis on graphs. *PhD Thesis, University of York*, 2007.
- [194] S. R. S. Varadhan On the behavior of the fundamental solution of the heat equation with variable coefficients. *Communications on Pure and Applied Mathematics*, 20(2):431-455, 1967.
- [195] S. T. Yau and R. M. Schoen. Differential geometry. *Science Publication*, 1988.
- [196] X. Bai, E. R. Hancock and R. C. Wilson. Geometric characterization and clustering of graphs using heat kernel embeddings. *Image and Vision Computing*, 28(6):1003-1021, 2010.
- [197] A. Grigor'yan. Heat kernels on manifolds, graphs and fractals. *Progress in Mathematics*, pp.393-406, 2001.
- [198] E. F. Moore. The shortest path through a maze. *In Proc. Internat. Symp. Switching Th., Part II. Cambridge. MA: Harvard University Press*, pp.285-292, 1959.
- [199] B. Luo. Statistical methods for point pattern matching. *PhD Thesis, University of York*, 2001.
- [200] Face database of the Max Planck Institute for biological cybernetics.  
<http://faces.kyb.tuebingen.mpg.de>
- [201] S. A. Nene, S. K. Nayar and H. Murase. Columbia Object Image Library (COIL-20). *Technical Report CUCS-005-96*, 1996.
- [202] CMU house sequence.  
<http://vasc.ri.cmu.edu/idb/html/motion/house/index.html>.



- [203] M. Gori, M. Maggini and L. Sarti. Graph matching using random walks. *ICPR*, pp.394-397, 2004.
- [204] R. C. Wilson and E. R. Hancock. Graph matching by discrete relaxation. *Pattern Recognition in Practice*, Vol.16, pp.165-176, 1994.
- [205] P. Ren, R. C. Wilson and E. R. Hancock. Pattern vectors from the Ihara zeta function. *ICPR*, pp.1-4, 2008.
- [206] L. Han, R. C. Wilson and E. R. Hancock. A supergraph-based generative model. *ICPR*, pp.1566-1569, 2010.