Interactive and Adaptive Visualisation For Mobile Data Mining

by

Hasnain AlTaiar

Thesis submitted in partial fulfilment of requirements for the Degree of Master of Information Technology with Honours in the Centre of Distributed Systems and Software Engineering at Monash University, Australia

October 2010

Declaration

I declare that the thesis contains no material that has been accepted for the award of any degree or diploma in any university and that, to the best of my knowledge, the thesis contains no material previously published or written by any other person except where due reference is made in the text.

Signed

Date

Centre of Distributed Systems and Software Engineering

Monash University

Caulfield East, Vic. 3145

Australia

Acknowledgements

I would like to thank my supervisor Dr. Shonali Krishnaswamy for her guidance, support, and invaluable ideas. I also thank Dr. Nicholas Nicoloudis, Dr. Mohamed Medhat Gaber, Brett Gillick, Abijat Sinha, and Jonathan Liono. Their support and encouragement are highly appreciated. Finally, I dedicate this thesis to my family and friends for their support and encouragement.

Interactive and Adaptive Visualisation For Mobile Data Mining

Abstract

There is an emerging focus on real-time data stream analysis and visualisation on mobile devices. The recent growth of smart phones capabilities has attracted many fields to port their applications to the mobile context. Examples of such mobile data analysis applications are patient monitoring, traffic monitoring, and stock market visualisation systems. Those types of applications require real-time analysis and visualisation of continuous, and voluminous datasets. Users of those systems need to make well-informed decisions in real-time based on the analysis of constant monitoring on different sensors or other data sources. Thus, mining data streams on mobile devices is becoming increasingly an important field that is termed as Mobile/Ubiquitous Data Mining (UDM).

UDM refers to the process of data analysis that is performed on portable and embedded devices in distributed environments where data is transmitted in form of a continuous stream. Users, who are in a need for data stream analysis and visualisation in time critical scenarios, are the main target of UDM visualisation since it allows data mining to be accessible anywhere at anytime. However, visualising the outcome of UDM analysis on mobile devices faces many challenges due to the resources constraints and the characteristics of display screens of mobile devices as well as the distinctive characteristics of data streams. While limited computational resources, small screen space, and limited power supply are some of the limitations of mobile devices, data streams also have challenging characterises such as being dynamic, changing, and continuous.

Several resource-aware and adaptive strategies and algorithms have recently been developed for UDM. This trend has shaded the lights on UDM visualisation as it is essential to quickly and appropriately understand the data analysis outcome. Clutter has been identified as a key issue in UDM visualisation, and clutter-adaptive techniques have also been developed. However, visualising the continuous analysis of real-time data streams also requires adaptive, interactive, and user-driven style of presentation to ensure accuracy and appropriate comprehension of the data visualisation for mobile decision-making.

Therefore, in this thesis, we propose, develop, and evaluate interactive, adaptive, generic, and user-driven/controlled UDM visualisation strategies that extend the state-of-the-art clutter-aware UDM visualisation strategies. We have implemented and experimentally evaluated the proposed framework to examine its effectiveness in terms of clutter reduction, visualisation enhancement, and improving resource management. The system was evaluated using publically available stock market data as well as geo-locations dataset that was obtained by geo-coding addresses from the web. Finally, we have provided demonstrations of the application, which has been made available on the web and on the Google Android App-Store for free download to show the operation of our system. The outcomes of this dissertation have resulted in a Demo paper in ICDM 2010, and a conference paper in ICTAI 2010.

Publications

- Gillick, B. AlTaiar, H. Liono, J. Krishnaswamy, S. Nicoloudis, N. Gaber, M. Sinha, A. 2010, "Clutter-Adaptive Visualisation for Mobile Data Mining" *Proceedings of The 10th IEEE International Conference on Data Mining*, ICDM 2010
- AlTaiar, H. Krishnaswamy, S. 2010, "User Interaction for Mobile/Ubiquitous Data Mining Visualisation" *Proceedings of the first international annual conference*, ARCHER 2010.
- Gillick, B. Gaber, M.M. Krishnaswamy, S. AlTaiar, H. Nicoloudis, N. Zaslavsky, A. Sinha, A. 2010, "Adaptive Visualisation Strategy for Mobile Data Mining", *Proceeding of the 7th International ICST Conference on Mobile and Ubiquitous Systems*, Mobiquitous 2010.
- Gaber, M.M. Krishnaswamy, S. Gillick, B. Nicoloudis, N. Liono, J. AlTaiar, H. Zaslavsky, A. 2010 "Adaptive Clutter-Aware Visualisation for Mobile Data Stream Mining", *Proceeding of the 22nd International Conference on Tools with Artificial Intelligence*, ICTAI 2010
- Haghighi, P.D. Burstein, F. AlTaiar, H. Arbon, P. Krishnaswamy, S. (2010), “Ontology-Based Service-Oriented Architecture for Emergency Management in Mass Gatherings” *Proceedings of the International Conference on Service-Oriented Computing and Applications,* SOCA, 2010

Table of Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

1.1 Preamble

Over the last few years, electronic data gathering and storage technologies have advanced significantly to increase the amount of data that we have exponentially. Increasingly, data mining and knowledge discovery need to develop highly scalable and fast techniques for coping with such large data volumes. Data Mining is defined as the use of sophisticated data analysis tools to seek interesting or valuable information within large datasets (Hand 1999). Data Mining aims to extract knowledge and identify valid relationships within voluminous data. Today, data mining is becoming more important as the amount of data has increased unprecedentedly and the speed of discovering interesting patterns is expected to be faster than ever (Han and Kamber 2006).

Massive datasets and the demand for analysing them are common in large number of application domains such as health, finance, and security. The financial domain offers many good examples of using data mining techniques to help investors to make better decisions. Enke and Thawornwong (Enke and Thawornwong 2005) mention that many applications in this domain have implemented a data mining mechanism to perform forecasting based on stock market data. Such applications provide investors with invaluable knowledge in order to make well-informed decisions.

An increasing focus is emerging on real-time analysis of continuous data streams. Besides, many applications have been presented to process and analyse data streams. These trends have led to a data stream phenomenon due to the many stream generators and the variety of applications that process these streams (Gaber and Yu 2006). As a result, the need for exploring these data streams and testing a specific hypothesis promoted the data stream mining field. Data stream mining is concerned “with extracting knowledge structures represented in models and patterns in non stopping streams of information” (Gaber, Zaslavsky et al. 2005).

In a parallel development, we have witnessed a remarkable growth in smart phones capabilities over the last few years. This growth has enabled bringing many of the data stream mining applications to the mobile phone environment (Gaber et al 2004). Examples of such applications are real-time moving cars monitoring (Gaber, Krishnasawamy, Gillick, Nicoloudis, Liono, and Zaslavsky, 2010) and mobile stock market visualiser (Kargupta, Park, Pittie, Liu, Kushraj, and Sarkar, 2002). These intelligent applications enable users to make better and well-informed decisions conveniently (anytime and anywhere). Processing and analysing continuous flow of data on portable devices is referred to as Ubiquitous Data Mining (UDM) (Liono 2009).

UDM is concerned with knowledge discovery through analysing continuous data streams on small portable devices in order to enable users to make decisions anywhere at anytime. Thus, mobile users in time critical environments are the main target of UDM. Given that all traditional data stream mining algorithms are designated for powerful computers, performing such task on mobile devices demands different approaches and techniques. Krishnaswamy et al. (Krishnaswamy, Gaber, Harbach, Hugues, Sinha, Gillick, Haghighi, and Zaslavsky, 2009) summarise the main challenges of UDM as resource constraints, temporal constraints, mobility, adaptation, as well as the streaming nature of the processed data. In this context, few UDM and UDM visualisation systems have already been developed such as CACV (Gaber, et al. 2010), MobiMine (Kargupta, et al. 2002), and VEDAS (Kargupta, Bhargava, Liu, Powers, Blair, Bushra, Dull, Sarkar, Klein, Vasa, and Handy, 2004).

In order to make a better understanding of the data analysis results, the UDM outcomes need to be visualised. Data Visualisation is defined as "graphical representation of information with the goal of helping the user to gain a qualitative understanding of the information" (Peng, Ward, and Rundensteiner, 2004). Bertin (Bertin 1981) added that visualisation refers to the process of formulating a mental model of the presented data by the user. Data stream mining visualisation is an emerging field that plays a key role in time critical applications, where users need to make informed decision in real-time based on the data analysis outcomes. Visualising data stream mining results on mobile phones brings with it many challenges including (Liono 2009):

- Limited screen size
- Limited power supply (small battery)

- Instant update of the data visualisation to reflect the data analysis process
- Dynamic and adaptive computation for visualisation
- Constrained resources such as memory, bandwidth, and processor speed

While the increasing computational capabilities of smart phones have helped in addressing some of the UDM hurdles, certain challenges like the small screen size and the limited energy supply will remain mitigating factors for UDM visualisation. Considering the very limited screen size of portable devices, the infinite nature of data streams, and the unpredictable results shape and size, visualising data stream mining on portable devices is a challenging task. Attempting to fit too much data in a very small display space can result overcrowded displays that are hard to read and understand (Ellis and Dix 2007). Thus, clutter reduction techniques are essential to reduce screen congestion in order to provide clearer and comprehendible visualisation (Nicholson and Vickers 2004).

There have been many clutter reduction techniques to reduce clutter and improve the data visualisation (Nicholson and Vickers 2004). Few of these techniques have already been leveraged for mobile data stream mining visualisation (Gaber et al. 2010). However, as established in the mobile Human Computer Interaction HCI (Burigat and Chittaro 2005; Burigat, Chittaro, and Parlato, 2008), interaction is an important factor that can improve visualisation, and promote better resource management practices. This thesis aims to propose, incorporate and evaluate interactivity for enhancing the UDM visualisation on smart phones.

1.2 Motivation and Objectives

The objective of this thesis is to develop an adaptive, interactive and generic data stream mining visualisation strategy that can be implemented on mobile devices. This visualisation strategy would reduce clutter, enhance visualisation, and promote better resource management.

As mentioned earlier, the recent significant growth in mobile devices capabilities has facilitated overcoming some of the data stream mining visualisation challenges. However, the small screen size constraint is one that is a result of the mobile nature of these devices, and thus it is not expected to be resolved by this growth. Some authors (Chittaro 2006; Burigat and Chittaro 2005) emphasised that developing applications for mobile devices without considering this constraint may result in a typical failure of graphs and figures that are hard to read and understand.

Currently, there are very few applications that have succeeded in analysing and visualising data on mobile devices such as Clutter Aware Clustering Visualiser CACV, Vedas, and MobiMini (Gaber et al. 2010; Krishnaswamy et al. 2009; Kargupta et al 2004; Kargupta et al 2002). However, none of these few applications in this area implements any means of interaction to reduce clutter or improve visualisation. Moreover, the available UDM visualisers overlook interactivity assuming that data is visualised effectively for all users and on different mobile phones. In reality, however, different mobile phones have different screen size, different resolution, and different colour sets. Also, different users have different level of data perception and clutter tolerance. Furthermore, when visualising data streaming, it is almost impossible to predict the UDM analysis results and size. Hence, given those facts, a graph could visualise the outcome of UDM effectively and accurately on one mobile phone for one user, but it is not guaranteed that it will do the same job on other mobile devices or for other users/applications (Bertin 1981; Chittaro 2006; Burigat and Chittaro 2005).

Additionally, there has been no systematic investigation that studies the implications of incorporating interactivity in mobile/ubiquitous data stream mining visualisation systems, although many researchers (Chittaro 2006; Burigat and Chittaro 2005; Burigat et al. 2008) have emphasised the importance of interaction in mobile visualisation applications. This can be reasoned to the emerging nature of this area (Gaber, Krishnaswamy, and Zaslavsky, 2005).

Hence, this thesis aims to enhance the mobile data stream mining visualisation through providing a generic, interactive, and adaptive data visualisation strategy and framework. The objectives of this research can be summarised as follows:

- Review the mobile data visualisation generally and the UDM visualisation in particular to determine and identify the main challenges of UDM visualisation and how can they be addressed.
- Propose and develop a generic, adaptive, and interactive UDM visualisation strategy that can minimise clutter, enhance visualisation, and promote better resource management practices.
- Implement and evaluate the proposed model to demonstrate the effectiveness and importance of interactivity in mobile data stream mining visualisation systems.

1.3 Dissertation Outline

This thesis is composed of six chapters, and organised as follows:

- Chapter 2 presents a comprehensive literature review regarding the mobile/ubiquitous data stream mining UDM. We review, analyse, and critique current techniques and algorithms of data mining generally and UDM in particular.
- Chapter 3 presents an overview of UDM visualisation from the mobile human computer interaction HCI perspective. We first look at the mobile data visualisation in general, and then move to UDM visualisation to determine the main challenges and the main approaches to address them.
- Chapter 4 presents our proposal for an interactive UDM visualisation model based on our findings from chapters 2 and 3. In this chapter, we present the theory, the algorithm and the framework of our proposed model.
- Chapter 5 presents the implementation and the evaluation of the proposed model. Firstly, the process of designing and implementing the system is discussed. Eventually, a comprehensive evaluation is conducted to assess the model's effectiveness based on different evaluation parameters.
- Chapter 6 shows the conclusions of this thesis by summarising our contribution and discussing our future research topics.

2 Review of UDM

2.1 Introduction

Recently, there has been an emerging focus on real-time data stream analysis on mobile devices. The phenomenal growth of mobile devices coupled with their ever-increasing computational capacity presents an exciting new opportunity for real-time, intelligent data analysis in pervasive/ubiquitous environments. However, performing data analysis and visualisation on portable devices still have many challenges that need to be addressed in order to satisfy the needs and expectations of modern mobile users. Currently, constrained resources, short battery life, and limited screen size are the main hurdles of mobile data stream mining and visualisation.

This chapter reviews the main topics of mobile/ubiquitous data stream mining. The chapter is organised as following:

- After this introduction, data stream mining is reviewed in section 2 to present the main data stream mining concepts and techniques.
- Mobile/ubiquitous data stream mining is presented in section 3.
- Finally, section 4 summaries our literature review in relation to mobile/ubiquitous data mining UDM. We draw our conclusions based on our findings from this review.

2.2 Data Stream Processing

The data acquisition hardware is advancing significantly as many life-related activities are getting digitized. Besides, many applications have been presented to process the continuous flow of data records. These trends have led to a data stream phenomenon due to the many stream generators and the variety of applications that process those streams (Gaber and Yu 2006). In general, there are two main forms of data stream processing, which are data stream querying and data stream mining (Gaber, Zaslavsky, and Krishnaswamy, 2005). In the next sections, we review both of the two approaches in order to understand the foundations of UDM.

2.2.1 Data Stream Querying

Recently, in many data-intensive applications such as financial applications, network monitoring, manufacturing, and others, data has been modelled best not as persistent relations but rather as transient data streams (Babcock, Babu, Datar, Motwani, and Widom, 2002). This is because data in such systems is continuous, rapid, time-varying, unpredictable, and potentially infinite (Babcock, et al. 2002). As a result, this causes serious fundamental problems to the traditional data management systems. For instance, it is possible that the arrival rate of data streams could be higher than the maximum throughput of the available disk storage (Babcock, et al. 2002).

Queries in general are classified into two main categories; one-time queries and continuous queries (Babcock, et al. 2002). The one-time queries category includes traditional database queries, which are evaluated once over a point-in-time snapshot of the dataset. On the other hand, continuous queries are evaluated continuously as data streams continue to arrive. In the context of data streams, continuous queries are the more interesting class of queries. Basically, the answer to a continuous query is generated over time, always reflecting the data stream that is seen so far. This answer might be stored and updated as new data items arrive, or it might be produced as data streams as well. Depending on the applications, one of those two approaches would be used (Babcock, et al. 2002).

Another distinction is between Predefined queries and Ad hoc queries (Babcock, et al. 2002). A predefined query is the one that is provided to the data stream management system before any data items has arrived. Generally, predefined queries are continuous queries, although scheduled one-time queries can also be predefined. Ad hoc queries, on the other hand, are supplied online after the data streams start to arrive. Ad hoc queries can be either one-time queries or continuous, and they are not known in advance, which complicates the design of the data stream management systems (Babcock, et al. 2002).

Overall, in data stream query processing, approximation (Duffield, and Grossglauser, 2000) and adaptivity (Avnur, and Hellerstein, 2000) are the key ingredients in executing queries and performing other type of processing over rapid data streams. In data stream mining, the focus is more on data stream mining than on the query processing. Therefore, the focus of the next section will be on data stream mining.

2.2.2 Data Stream Mining

Data stream mining is concerned with “extracting knowledge structures represented in models and patterns in non stopping streams of information” (Gaber, Zaslavsky et al. 2005). This was promoted by the need for exploring data streams and testing a specific hypothesis in those rapidly and continuously coming data items.

Generally speaking, although data steam mining is a sub-field from the traditional data mining, it follows different approaches for achieving the same goal of knowledge discovery (Liono 2009). There are many reasons for that, yet the main difference lies in the nature of the processed data in the two fields. In other words, mining a data stream involves analysing and processing a set of data that is flowing continuously in real-time with a very little chance of storage capability.

Unlike in data stream mining approach, traditional data mining usually involves intelligent analysis on large datasets that are extracted from persistent data storage. This difference in the nature of the processed data in data mining and data stream mining systems enforces the latter to follow different innovative techniques to achieve the discovery of important information and patterns from the data streams (Han and Kamber 2006). These data streams could be coming from a variety of sources such as real-time patient monitoring, sensor networks, security monitoring systems, or other applications. Therefore, almost always, the data analysis process must be accomplished in real-time with a very little chance of storing that data for offline processing.

In addition to the difference in terms of data persistency, data streams have many different characteristics than the ordinary data. Examples of these characteristics are "temporarily ordered, fast changing, massive, and potentially infinite" (Han and Kamber 2006, PP.468). The fact that data streams are "temporal" challenges the traditional data mining strategies, which are designed to deal with persistent data. Hence, data stream mining requires sophisticated techniques that can test hypothesis and discover important patterns in real-time, while the data streams are in memory. To make it more complicated, data streams are also characterised as "massive" and "potentially infinite". This contributes to the whole complexity since data stream mining requires dealing with data in real-time while it is in memory, yet at the same time the amount of data is "massive" and "potentially infinite", which could cause the system to crash when it runs out of memory. As a result, data minimisation strategies are seen as essential to data stream mining. Data summarisation is one of the examples of data minimisation techniques (Liono 2009).

2.2.3 Data Stream Mining Techniques

Since data stream mining is an extended field from the data mining area, most of its techniques are derived from the traditional data mining algorithms. In general, data stream mining techniques are classified into three main types, which are Clustering, Classification, and Frequency counting (Gaber, Zaslavsky et al. 2005).

Clustering is a very well known technique that is used to uncover structures that were not known previously within a large dataset. It refers to the process of grouping sets of data items to clusters (Barbara' 2002). Each cluster has a set of objects that share similar attributes. In other words, objects will be assigned to clusters depending on their similarities and differences. Besides, clusters share some similar attributes such as visual representations, but they are differentiable by some other distinctive characteristics such as centre point or centroid (Guha, Mishra, Motwani, and O'Callaghan, 2000).

To be applicable in any system, clustering technique requires some technical characteristics in that system. These requirements are listed by (Barbara' 2002) as compactness of representation, fast incremental processing of new objects, and clear and quick identification of “outliners”. These requirements reflect the distinctive characteristics of data streams that are mentioned earlier. For instance, since data streams flow continuously, there is usually no realistic system that can afford analysing these lengthy and “potentially infinite” data streams. Therefore, for an algorithm to cluster data streams, it must provide a compact representation of available clusters. Also, this representation should not grow linearly with the number of points. That is to reduce the use of memory in these data stream mining systems as much as possible. While the fast incremental processing requirement is far from being a trivial condition given the need for on-line placement of new objects, the identification of “outliners” is the least intuitive character among the others. Identification of “outliners” refers to the ability of recognising points that do not fit well in any cluster. Considering the dynamic nature of data streams, data stream mining algorithms need to efficiently identify these “outliner” points and decide where to place them (Barbara 2002).

On the other hand, Aggarwal (Aggarwal 2006) emphasized that it is difficult to adapt arbitrary clustering algorithms to data streams. That is because of the difficulty of relating each new object to one of the present clusters. Thus, some clustering techniques that are based on random and approximation solutions have emerged to help in clustering the new coming objects. K-mean is one of the very popular clustering algorithms, which uses mean-based analysis (Ordonez 2003). Another common example of clustering algorithms is the K-median, which is a median-based analysis (Guha, Mishra et al. 2000).

The second main data stream mining technique is Classification, which means mapping incoming data items into one of several pre-defined categories (Gaber, et al. 2005). Like clustering, the classification algorithm categorises data objects based on their attributes. Technically speaking, the classification algorithm includes two different phases, which are model building and data classification (Han and Kamber 2006). Firstly, the system would have to construct a classification model, which establishes class labels and data structures. Eventually, this classification model would be used for dynamically classifying the incoming items depending on the structure of the previously built model. While this algorithm is widely used in the traditional data mining systems, the nature of data streaming demands innovative solutions for applying such algorithms on data streams to avoid the multiple data scans. Moreover, in order for a classification algorithm to categorise data streams, it must be dynamically adaptive to the rapidly changing patterns of data streams (Liono 2009).

Basically, Wang et al. (Wang, Fan, Yu, and Han, 2003) identified three main requirements for classifying data streams. These requirements are accuracy, efficiency, and ease of use. As for all other algorithms, accuracy is always an important issue, yet it is more important for the classification algorithm because of the rapidly evolving structure of data streams. It is concerned that classifying a massive number of records could drop the accuracy of the classification algorithm (Aggarwal, Han, and Yu, 2004). Liono (Liono 2009) mentioned that a good solution could be keeping the data history to achieve a constant high accuracy.

Frequency Counting, which is the third main data mining technique, refers to the process of counting the frequent occurrence of items in a dataset (Liono 2009). Giannella et al. (Giannella, Han, Pei, Yan, and Yu, 2003) mentioned that although frequent-pattern mining was studied widely on ordinary data, it is still challenging to extend it to data streams. They reason that to the fact that data streams have far more information to track and significantly more complex to mange. A study by Giannella et al. (Giannella, Han et al. 2003) proposed a prototype for computing and maintaining frequent patterns in an FP-Stream (which is smaller and more stable than data streams).

Clustering, classifying, and frequency counting are all data stream mining techniques, yet they have different approaches and algorithms to discover knowledge and analyse data streams. Generally, these approaches can be categorised into two main groups, which are data based and task based (Gaber, Zaslavsky et al. 2005). While the data based approaches are concerned with the subset representation of the original dataset, the task based approaches investigate the performance/outcome of data stream mining operation to achieve efficiency in terms of time and space. Sampling, sliding, load shedding, sketching, synopsis and aggregation are categorised under data based approaches. Task based approaches include sliding window, approximation algorithms, and algorithm output granularity. In addition, many other approaches could be added to these categorisations such as multi-resolution method and histograms (Han and Kamber 2006). In the following sections, we briefly describe some of these approaches in relation to our focus in mobile/ubiquitous data stream mining.

To start with, Sampling is a technique of randomly selecting data items to represent a time series dataset in the data stream mining process (Han and Kamber 2006). It is widely used since it is almost impossible to analyse a “potentially infinite” stream of data. In order to obtain accurate and unbiased samples of data, Han and Kamber (Han and Kamber 2006) demanded that in a periodic random sampling, the length of the data stream should be known. Gaber et al (Gaber, Zaslavsky et al. 2005), therefore, pointed out that using sampling mechanism in data stream mining might have some accuracy issues since data streams have no known or predictable size.

In the context of data stream mining, load shedding is defined as a technique of “dropping unprocessed tuples to reduce system load” (Aggarwal 2006). Liono (Liono 2009) mentioned that it is difficult to implement the load shedding technique since it drops the sequence of data that could be potentially significant for pattern recognition operations.

Aggarwal (Aggarwal 2006) defines Sketching as a data stream mining technique that project datasets randomly. Basically, sketching technique produces sketches of samples that are picked randomly from a dataset. Thus, while Aggarwal (Aggarwal 2006) mentioned that sketching is extremely efficient, he and some others (Liono 2009) pointed out that inaccuracy is a major issue of this technique since it uses sampling of datasets.

Due to the infeasibility of storing the complete streams, some innovative techniques have emerged to present a summary structure of datasets such as synopses (Golab and Ozsu 2003). Synopsis is an approximate data structure that summarises a dataset or a data stream in order to be used for data stream mining analysis (Liono 2009). In general, it uses approximation to compute the frequency of items in data streams. One example of synopsis is Histogram, which distributes data items among different containers that have fixed size.

On the other hand, task based approaches are concerned with the performance of the analysis process. Basically, these approaches use approximation algorithms to obtain higher performance in solving complex computational problems due to the constrained resources. Therefore, the main challenge would be in designing an efficient approximation algorithm. Sliding window is an example of the task based approaches. It performs analysis over the most recent data streams by continuously processing the whole stream in one scan. To put it simpler, assuming a data record arrives at time *(t)* with a fixed window size *(w)*, the algorithm will analyse the data stream of the time interval between t and *(t+w)* (Han and Kamber 2006). In fact, sliding window uses the summary of the previous data records with the most recent ones to do the analysis. For that reason, it is seen as an adequate solution for applications that require real-time analysis such as sensor networks (Liono 2009).

Another example of task based approaches is multiresolution, as mentioned by (Han and Kamber 2006). Multiresolution can be seen as a possible solution for data stream mining systems that are expected to receive potentially infinite stream of records by performing data reduction. Using divide and conquer mechanisms, the multiresolution technique builds a multiresolution data structure that allows multilevel analysis on data streams. Basically, a hierarchical structure - such as tree - is usually used for multiresolution data structure (Liono 2009).

Algorithm Output Granularity (AOG) is a resource-aware data stream mining technique that analyses data based on the available local resources and processing speed (Gaber, Zaslavsky et al. 2005). According to Gaber et al. (Gaber, Zaslavsky et al. 2005), AOG is divided into three stages; Data mining, adaptation, and merging. The system adapts to the currently available resources and data stream rates after the data mining stage. The system will then proceed to the merging stage when most memory spaces are occupied. At merging stage, the system merges, summarises and simplifies the knowledge structure of the data streams (Gaber, Zaslavsky et al. 2005; Gaber, Krishnaswamy, and Zaslavsky, 2004).

In summary, this section has outlined some of the main concepts, challenges, and techniques of data stream mining. We now review the area of performing data stream mining in portable environments, which is called Mobile/Ubiquitous data stream mining. Many recent studies have been conducted in this area in order to enable performing data stream mining on portable devices. Thus, this topic will be our main focus for the next section.

2.3 Mobile/Ubiquitous Data Stream Mining

Mobile/ubiquitous data mining (UDM) is defined as the "process of data analysis performed on mobile, embedded and ubiquitous devices" (Gaber, Krishnaswamy et al. 2004). It aims is to discover knowledge and analyse data on small portable devices to enable users to make decisions anywhere at anytime. UDM has a key role in time critical scenarios, where users need to comprehend massive amount of data quickly and make decisions accordingly. As explained earlier, performing such task on mobile devices demands different approaches and techniques. The recent growth in mobile devices capabilities has enabled such tasks on smart phones. This section reviews the challenges and techniques of UDM as well as the most recent systems in this area.

2.3.1 UDM Challenges

Performing UDM analysis on smart phones raises many challenges. While some of these challenges are related to the nature of the portable devices, some others are due to the data stream's characteristics. Thus, processing those streams requires dynamically adaptive algorithms that can handle those rapidly evolving data streams.

On the other hand, due to their compact and portable nature, mobile devices usually have constrained resources such as memory, processor speed, and battery power (Gaber, Krishnaswamy et al. 2010). Moreover, Gaber et al. (Gaber, Krishnaswamy et al. 2010) lists more challenges such as mobility of users, connectivity issues, and dynamic adaptation to the level of available resources. As a result, performing data stream mining on smart phones requires innovative and computationally efficient algorithms and techniques.

2.3.2 UDM Techniques

As mentioned earlier, performing data stream mining in ubiquitous environments requires adaptive algorithms and computationally efficient operations in order to cope with the high rate of incoming data records in the resources-constrained devices. Gaber et al. (Gaber, Krishnaswamy et al. 2003) proposed "Algorithm Output Granularity" (AOG) approach that uses resources-awareness to adapt to the available memory, time constraints, and data stream rate. Basically, AOG uses the available memory to perform data mining process on portable devices. Once the memory is full, the algorithm would then switch to the knowledge integration process, which is basically merging output results of the data mining operation. This can be seen in figure 2.1.

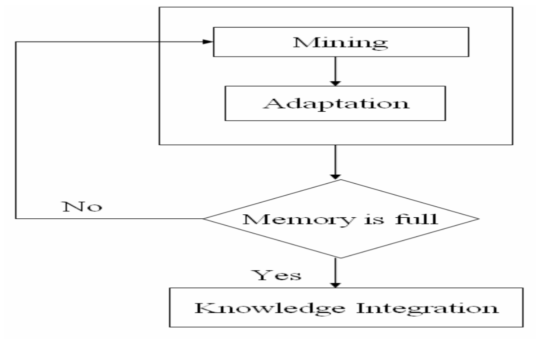


Figure 2.0.1 AOG Process (Liono 2009)

Gaber et al. (Gaber, Krishnaswamy et al. 2003) mentioned that AOG is a generic adaptive algorithm that is applicable to clustering, classification and frequency counting techniques. In fact, using the AOG approach, Gaber et al. (Gaber, Krishnaswamy et al. 2003) developed many data analysis algorithms that can run in ubiquitous environments such as Light Weight Clustering (LWC), Light Weight Classification (LWClass), and Light Weight Frequency count (LWF) (Liono 2009). Additionally, Shah et al. (Shah, Krishnaswamy, and Gaber, 2005) developed an algorithm that can also be used for mobile/ubiquitous data mining, which they called Resource-Aware Very Fast K-Mean (RA-VFKM). Gaber et al. (Gaber, Krishnaswamy et al. 2004) said that LWC, which uses the AOG approach, outperforms the K-means algorithm in terms of processing time while it maintains similar accuracy. Hence, the state-of-the-art CACV (Clutter-Aware Clustering Visualiser) system uses the light weight clustering algorithm (LWC) (Gaber, Krishnaswamy et al. 2010).

Generally speaking, light weight clustering LWC is a sophisticated algorithm that is based on intelligent techniques to enable mobile/ubiquitous data mining (Gaber 2009). Basically, it uses single-scan technique to quickly analyse high-speed data streams by placing the new data items in the relevant cluster in memory. Once the memory is full, knowledge integration process starts by incrementally integrating clusters to free some memory spaces (Liono 2009).

Inaccuracy issue is seen as a major concern in classification algorithm because of the data streams' evolving nature where patterns are changing rapidly over time. For solving that, historical data is required to maintain higher accuracy. However, Aggarwal et al. (Aggarwal, Han et al. 2004) mentioned that it is very expensive to keep the entire history of data streams. Thus, others (Gaber, Krishnaswamy et al. 2003; Shah, Krishnaswamy, et al 2005) suggested summarising the historical data as a solution.

The light weight classification algorithm, on the other hand, is based on the concept of nearest neighbour. Basically, it classifies the new incoming data item by finding the nearest instance in memory, which is found by comparing the distances from other instances. Those instances usually have some attributes that determine the status of each instance. For example, instances could be kept or removed from memory depending on their weight attribute, which could be increasing or decreasing according to the number of items in each instance (Gaber, Krishnaswamy et al. 2003).

The Light Weight Frequent Items (LWF) is another AOG-based algorithm that enables data stream mining in mobile environments. It determines the maximum number of data records to be analysed by using counters. Once all counters are occupied, if it receives a new distinctive data object, the algorithm would then just ignore that item and decrease all counters by one. After a predefined time period, the algorithm removes the least frequent items and reset their counters to zero (Liono 2009).

The last three algorithms (LWC, LWClass, and LWF) have shown a good level of performance and accuracy by managing resources efficiently and consistently (Gaber 2009).

2.3.3 UDM Systems

Considering the UDM issues as above, new innovative techniques are required to enable mobile users to make use of their portable devices to analyse and visualise data. With the recent growth in their capabilities, smart phones are capable of performing such tasks onboard. In this context, (Krishnaswamy, Gaber et al. 2009) developed the first generic toolkit for mobile data mining, which they called “Open Mobile Miner”. OMM is a toolkit that can be deployed on a range of mobile devices in order to enable mobile/ubiquitous data mining.

Recently, smart phones have attracted many researchers to leverage the existing data stream mining techniques to enable their operation on those devices. Some studies have resulted efficient and accurate data stream mining algorithms on handheld devices such as Personal Digital Assistants PDAs (Kargupta, Park et al. 2002; Kargupta, Bhargava et al. 2004) and smart phones (Gaber, Krishnaswamy et al 2005; Gillick, Krishnaswamy et al. 2006; Gaber, Krishnaswamy et al 2010). For instance, an innovative client/server data stream mining system named MobiMine was developed by Kargupta et al, which focuses on stock market data mining on PDAs (Kargupta, Park et al. 2002). Additionally, Kargupta et al. also developed an application that allows continuous monitoring and pattern extraction onboard a moving vehicle. This was called Vehicle Data Stream Mining System VEDAS (Kargupta, Bhargava et al. 2004).

MOLEC is another example of mobile data stream mining systems that was developed by (Rodrigues, Goni, and Illarramendi, 2005). MOLEC stands for “Monitorización On-Line de Enfermos del Corazón” (On-Line Monitoring for Heart Patients) (Rodrigues, Goni et al. 2005). It enables cardiac monitoring through analysing ECG signals in order to identify a range of anomalies and arrhythrnias. Indeed, MOLEC uses an off-line generated decision tree to perform the data analysis task.

Furthermore, Gaber et al. (Gaber, Krishnaswamy et al. 2010) proposed an innovative application that was called Clutter-Aware Clustering Visualiser (CACV), which incorporates resource-awareness and clutter-awareness techniques. While the resource-awareness is necessary for adapting the algorithm dynamically to the currently available resources, clutter-awareness is needed for reducing the amount of clutter on the limited screen space. By understanding the key challenges of mobile data stream mining visualisation such as constrained resources and limited screen space, CACV uses innovative techniques to cope with those challenges. These techniques are covered thoroughly in the next chapter.

In this section, we have reviewed concepts, challenges, and techniques of data stream mining in the mobile/ubiquitous environments. We have also reviewed the main recent projects in this area, which have motivated our research.

2.4 Conclusion

This chapter has reviewed the main concepts, techniques and approaches of data mining in general and UDM in particular. As explained earlier, data mining are generally classified into three main categories; clustering, classifying, and frequency counting. We have presented a comprehensive review along with examples of each one of these categories. We have also shown the main characteristics of data streams that enforce UDM to follow different approaches. These characteristics include "continuous, rapidly changing, massive, and potentially infinite".

Recent studies in UDM analysis were reviewed in this chapter. Many of the new UDM analysis algorithms were presented along with an overview of each one of them. Based on our literature review, Algorithm Output Granularity seems to be the state-of-the-art approach in UDM analysis since it uses resource-awareness to cope with the constrained resources. Some of the applications that demonstrate the effectiveness of AOG approach (such as CACV) were reviewed briefly. When comparing CACV system with the other available UDM applications, CACV takes the lead because of its resource-awareness, energy-awareness, and the clutter reduction techniques. It is also worth mentioning that CACV is the only application among the reviewed ones that supports sophisticated visualisation mechanisms when visualising the UDM outcomes.

Finally, since there are now UDM algorithms, which are operational on mobile devices, we conclude that the focus should be shifted to how to improve the operation and performance of these algorithms in the mobile context.

Performing the UDM analysis on the portable devices is only one part of the UDM visualisation. The second main requirement is to visualise the outcomes of the UDM process to enable users to make better and well-informed decisions. Thus, in order to better understand the status of the current UDM visualisation systems and how can they be improved, the next chapter reviews the mobile visualisation, and the UDM visualisation techniques, challenges, and possible solutions.

3 Review of UDM Visualisation

3.1 Introduction

UDM visualisation is an emerging topic that is part of the general mobile data visualisation field. Data visualisation is defined as a "graphical representation of information with the goal of helping the user to gain a qualitative understanding of the information" (Peng, Ward et al. 2004). Bertin (Bertin 1981) added that visualisation refers to the process of formulating a mental model of the presented data. Charts, graphs, maps are more meaningful for users when compared to numerical representations and they help in making well-informed decisions quickly and more conveniently, especially in real-time. This is because it is easier and faster for humans to comprehend/interpret visual graphics than to read text or numbers.

In order to identify and address the main challenges of UDM visualisation, we first need to study the strategies and the general principles of mobile data visualisation. By identifying the main requirements and the general approaches of mobile data visualisation, we can then determine what is required in UDM visualisation and the ways to achieving that. Thus, in this chapter, we first review the mobile visualisation strategies and principles. After identifying a set of typical requirements for mobile visualisation, we aim to use these as guiding principles that underpin our proposed contribution to UDM visualisation.

This chapter is organised as follows:

- Section 2 presents the principles and strategies of mobile data visualisation. This is important to understand the foundations of mobile visualisation in general.
- Section 3 presents a comprehensive review in the literature regarding UDM visualisation. We identify UDM visualisation challenges and essentials based on our review of mobile data visualisation.
- Section 4 presents an overview of a state-of-the-art UDM visualisation application, namely CACV (Clutter-Aware Clustering Visualiser), which we aim to extend/enhance in this thesis.

- Finally, conclusions and a summary of the UDM visualisation challenges and essentials are presented in section 5.

3.2 Data Visualisation Strategies and Principles

Information visualisation involves the use of computer(s) to represent abstract data visually and interactively to the user to amplify cognition (Card, Mackinlay, and Schneiderman, 1999). Bertin (Bertin 1981) also explained that a graphic is "no longer 'drawn' once and for all". Instead, it is 'constructed' and 'reconstructed' (manipulated), and 'reconstructed', until the relationships, which lie within it, have been perceived. Additionally, he (Bertin 1981) added that a graph is never an end in itself and it is a moment in the process of decision-making. This concept is especially correct when visualising a rapidly changing stream of data items or their analysis as in the case of UDM visualisation. Finally, Spence (Spence 2001) reviews visualisation as a "human cognitive activity not something that the computer does". This shows how interaction is tightly coupled with visualisation, which means that in order to make a visualised data comprehendible, users must have controls to interact, select, and adjust the presented data (McCrickard, Catrambone, Chewar, and Stasko, 2003).

On the other hand, offering too much information can be distractive or dangerous to the process of decision-making (Campbell, and Tarasewich, 2004). Albeit, users must still be given the option to interact with the visualised information to get all the contents when they want and based on their circumstances. Moreover, Burigat and Chittaro (Burigat and Chittaro 2005) emphasised the same approach, yet they termed it as "details-on-demand" functionality. "Detail-on-demand" refers to providing the user with the capability to obtain detail information about any element on the screen at any point in time by tapping/touching/clicking that element (Burigat and Chittaro 2005). Other studies (Chittaro, 2006; Burigat, Chittaro, and Parlato, 2008) also agreed about the importance of such techniques, but they warned that hiding more information and making them available on an event (click/tap/touch) might still cause data misinterpretation. Thus, they suggested that data visualisation systems should provide as brief information as enough for a user to make sense of the visualised data.

Since mobile data visualisation is different from data visualisation on desktop computers, we devote the next section to mobile data visualisation to review the differences, challenges, and approaches for visualising data on such constrained platforms.

3.2.1 Mobile Visualisation

Visualisation on mobile devices requires different techniques and approaches in order to cope with the many limitations that are enforced by the mobile context (Burigat and Chittaro 2005; Burigat, Chittaro et al. 2008; Tonder and Wesson 2008). While the small size of the screen is seen as a key challenge for data visualisation on portable devices, mobiles have many other limitations. Those constraints summarised by (Encarnacao, Frühauf, et al. 1995; Chittaro 2006; Tonder and Wesson 2008) as follows:

- Display constraints such as small screen, low resolution, and less colours.
- Limited computational resources (Processor, memory, graphic hardware).
- I/O peripherals are not suitable for complex tasks.
- No standardised mechanisms of interface. Each mobile device has its own specific interaction mechanisms.
- Limited energy resources (usually mobile phones hindered with small batteries).
- Connectivity issues in the mobile context.
- Lack of support for high-level graphic libraries (although this is changing notably with the emergence of today's smart phones)

For visualising data in an environment that has all the above-mentioned limitations, data visualisation techniques would need to be more innovative and interactive. In fact, (Chittaro 2006) mentioned that developing visualisation applications without considering the special requirements of these constrained platforms will result typical failure in graphs and charts that are hard to read and understand. Many researchers (Burigat and Chittaro 2005; Burigat, Chittaro et al. 2006; Chittaro 2006) agreed that since mobile data visualisation is totally an emerging field, there are currently no typical guidelines or successful-proven practices to follow in this area. Chittaro

(Chittaro 2006), however, listed some key characters as general discipline for mobile data visualisation systems. These characters are:

1. **Mapping:** showing all the important information in shape of objects (lines, circles etc) in a way that explains their values and the relationships among them.
2. **Selection:** users must have the ability to choose what is relevant to their current tasks. Showing less information could lead to taking suboptimal or plainly wrong decisions. Also, providing so much information could distract the attention from the important data and makes it harder to reason the problem in hands.
3. **Presentation:** this refers to the way, in which that data is presented to the user, which is an important issue that many applications fail to address. Presentation here refers to the way that information is laid out on the available screen space.
4. **Interactivity:** This refers to the importance of providing tools to rearrange and explore the visualised data, which is crucial in mobile data visualisation applications. Visualisation is tightly coupled with interactivity, and thus users must have different means of interaction to explore, navigate, personalise, and adjust the presented data.
5. **Human Factors:** basic knowledge of the differences in peoples' perception and cognitive capabilities is needed to design better applications that can suite different people. For a data visualisation to be successful, it must amplify the mobile users cognition, and this will not happen unless different aspects of human perception of data are taken into account when developing mobile visualisation applications.
6. **Evaluation:** after visualising the data, the system should be tested on real users to get feedback. Evaluating data visualisation can be very difficult and complex task. That is because different users have different level of data perception and this can also vary based on the type of dataset, type of application, mobile device, context in which the visualisation is happening, and many other factors. Generally, there are two main approaches for evaluating data visualisation systems, either laboratory based evaluation, or field based. We discuss this in more details later.

Moreover, visualising continuous - rapidly changing - data streams requires remarkable amount of computation and reasonably enough screen space. Since these two requirements are key constraints of mobile phones, many authors (Burigat and Chittaro 2005; Tonder and Wesson 2008; Gaber, Krishnaswamy et al. 2010) mentioned that dynamic adaptation could be a good solution for this issue. Thus, many techniques were presented to effectively use the limited screen space and resources to visualise data streams on smart phones.

Additionally, since the IO peripherals of mobile devices are notably limited, user interface has to be simplified as much as possible to make it easier for the user to interact, personalise, and comprehend the presented data (Karstens, Kreuseler, and Schumann, 2003). For instance, mobile users have no mouse or a physical keyboard that can help them to interact with mobile devices. Portable devices are normally compacted and small in size so that to facilitate carrying and moving them. Therefore, the visualisation applications must be very adaptive, interactive, with simple and easy to use interaction controls.

On the other hand, producing efficient and accurate visualisation of complex data (streams) requires high adaptivity, which means higher frame rates (fast display updates) and more computational operations (Karstens, et al. 2003). This computational overhead, according to (Karstens, et al. 2003), can be reduced by (1) using 2D and simple primitive shapes, (2) restricting the number of displayed items at a time and (3) simplifying the output figures. This is especially suitable in the case of visualising the analysis results of continuous data streams. In addition to this gain in terms of improving the resource management, limiting the number of the visualised items and establishing appropriate mechanisms for selecting the visualised items can improve the overall data visualisation (Karstens, et al. 2003). This has to be coupled with intuitive mechanisms for accessing the hidden information.

As a result, having appropriate mechanisms for limiting the number of visualised clusters/items/objects, with establishing intuitive techniques that enable mobile users to access the full contents of the visualised data, can improve data visualisation, reduce resource consumption, and enhance resource managements (Karstens, et al. 2003).

When visualising a large number of objects (as in the case of data streams), it is expected that some of the objects will be out of the scope of the screen, which are normally called off-screen objects (Burigat, et al. 2006). Visualising those off-screen objects is important for making the user aware of the global image of the visualised data. It is required that users should be aware of those off-screen objects to have a comprehensive and appropriate understanding of the overall results. However, visualising those objects can be very difficult since portable devices have limited screen space, which gets cluttered very quickly with visualising only the on-screen objects. Few techniques have already been developed for making the user aware of those unseen objects (Burigat, et al. 2006). These techniques include Halo, Arrows, and CityLight (Burigat, et al. 2006; Trevor, et al. 2001;Buyukkokten, et al. 2000; Björk, et al. 1999). While some of those techniques have shown successful operation on mobile devices, they are still error-prone since they only provide approximate indicators to the locations of the off-screen objects (Burigat, et al. 2006). Also, these techniques involve adding signs, lines, or shapes to alert the user about the off-screen objects, which could contribute to the clutter on screen. Some authors (Charaniya, and Lodha, 2003) suggested using different means of communication channels to alert mobile users about such items, this was called "speech-based visualisation" as we explain in the coming section.

In terms of evaluating the mobile data visualisation, Kjeldskov and Graham (Kjeldskov, and Graham, 2003) studied a large number of mobile HCI publications and they concluded that huge portion of that research was based only on laboratory evaluation and thus they demanded that field studies and real-user surveys type of evaluation are required. However, in response to this study (Kjeldskov, and Graham, 2003), Kjeldskov, et al. (Kjeldskov, Skov, Als, and Høegh, 2004) conducted a comprehensive research to find the difference between evaluating mobile applications in laboratory and the field-based evaluation. Comparing the lab-based evaluation-results of 6 studies to the field-based evaluation-results of the same cases, the authors concluded that the difference is very little and thus it does not worth the hassle and the cost of having the field-based evaluation. (Kjeldskov, et al. 2004).

Finally, according to (Paelke, Reimann, and Rosenbach, 2003), poor visual presentation of data and faulty implementation of interaction in mobile applications are key problems in designing interfaces for mobile applications. Burigat and Chittaro (Chittaro

2006; Burigat and Chittaro 2006) agreed that currently there are no generic and adaptive mobile data visualisation strategies.

3.2.2 Clutter Reduction Techniques

Mobile phones normally have a small screen size, which could be quickly occupied with information. Thus, clutter is one of the main hurdles that data visualisation faces on smart phones. Peng et al. (Peng, Ward et al. 2004) define clutter as overcrowded and disordered visual profiles that cause the structure of visual presentation of information to be ambiguous. In general, clutter is most likely to appear in the data stream visualisation systems due to the rapidly increasing number of data items or their analysis results that need to be visualised. Fuchs and Schumann (Fuchs and Schumann 2006) mentioned that clutter is a consequence of visualising too much information on small screen space. Therefore, they suggested that special consideration should be given to the effectiveness and expensiveness of data items visualised on that limited space.

Recently, many clutter reduction techniques have been presented to reduce clutter. According to (Ellis and Dix 2007), there are three main categorisations of clutter reduction techniques. The first category is called appearance, which includes techniques that affect the data visualisation appearance. Examples of this category are sampling, change point size, filtering, clustering, and change opacity. Clustering, for instance, is regarded as appearance technique since it reduces the amount of clutter by grouping different lines or points together in clusters. While both filtering and sampling techniques control clutter by reducing the number of data items to be visualised, they are different in the way they select the data records. Generally, sampling is based on a random selection method, whereas filtering technique selects a number of data items depending on a certain predefined criteria.

Spatial distortion is the second category of the clutter reduction techniques, which includes approaches that reduce clutter by displacement of points or lines in some patterns. Zooming and fish eye are widely used techniques of this category. Basically, techniques in this category involve many operations such as point/line displacement, space-filling, topological distortion, pixel plotting, and dimensional recording. In this context, topological distortion refers to the mechanism of changing the spatial view by stretching the background to differentiate each data item on screen. Also, the technique of representing each data item as a pixel and visualising it on the screen is referred to as Pixel plotting technique (Ellis and Dix 2007; Liono 2009).

On the other hand, the space-filling technique is usually used to solve the issue of overlapping data items by rearranging the visualised data items based on a certain data structure. Moreover, adjusting the data items (points/lines) alignment when displayed on screen is known as Point/Line displacement. For multi-dimensional data visualisation, some of these techniques like point/line displacement are used in multi-dimensional order alongside with dimensional reordering, which is another clutter reduction technique. In the third category of clutter reduction techniques, which is called temporal reduction, reducing the clutter in certain period of time is used as a strategy to control clutter on screen. Using animation for visualising data items on mobile phones could be a good example of temporal reduction techniques.

According to (Ellis and Dix 2007), clutter reduction techniques have several requirements in order to achieve the aim of presenting understandable information. For instance, clutter reduction techniques should avoid overlapping and keep the spatial information comprehendible. Scalability, adjustability, the ability to show point/line attribute(s), data item distinctiveness, and the ability to identify overlap density, are all some of the other requirements of clutter reduction techniques.

3.2.3 Speech-based Visualisation

Charaniya and Lodha (Charaniya and Lodha 2003) introduced the use of the speech as an effective mean of communicating with the machine. There have been several libraries that support voice recognition and the conversion of the plain text into speech, which is normally termed as Speech Synthesis (Charaniya and Lodha 2003). Many of these libraries are currently supported on the new smart phones such as Nokia[1], Google Android[2], and Microsoft Mobile 7[3]. Speech synthesis is defined as the "process of generating an audio signal from a given text, it is also referred to as Text-to-Speech

[1] Nokia text-to-speech support http://www.nokiausa.com/get-support-and-software/software/text-to-speech
[2] Google Android text-to-speech support http://code.google.com/p/eyes-free/wiki/TTSLibraryExplanation
[3] Microsoft Speech library http://www.microsoft.com/downloads/en/details.aspx?FamilyID=5e86ec97-40a7-453f-b0ee-6583171b4530&displaylang=en

(TTS)" (Charaniya and Lodha 2003). Broadly, there are two main categories of speech synthesis systems, restricted and unrestricted text to speech systems (Charaniya and Lodha 2003). While restricted TTS systems have only limited number of words such as numbers and dates, unrestricted TTS refers to converting the generally used words and sentences into speech. In other words, unrestricted TTS libraries can be used on smart phones to give feedback to mobile users by enabling the handheld devices to read the alert message(s) to the user when an alarming trend happens.

The use of synthesis speech to enable effective interaction between mobile devices and users has shown impressive results that promise to bring many advantages to the mobile visualisation applications (Charaniya and Lodha 2003). In a case study (Charaniya and Lodha 2003) about using the speech as an input/output tool in combination with visual aids and without visuals, results showed that users were notably advantaged by the use of synthesis speech. In both cases (with visuals and without), the use of speech showed a positive impact in terms of helping users to better understand the output as well as to ease the use of the application.

Moreover, (Campbell, and Tarasewich, 2004) pointed out that humans have a "limited amount of resources available for allocation to different tasks" and so they cannot "attend to everything at once". Holland and Morse (Holland and Morse, 2001) added that a mobile application might not be the focal point of the user's current activities. Besides, Johnson (Johnson, 1998) also mentioned that the amount of attention a user can give to a mobile application will vary over time, and users' priorities can also change quickly and unpredictably. This is especially true for mobile users when deployed to the field and they are expected to handle many tasks at once. Overloading the users' attention with many notifications and attention-demanding type of visualisation "may prove to be ineffective or ignored completely" (Campbell, and Tarasewich, 2004). Hence, it is required that mobile applications must be ensured not to overload the attention of the intended recipient by the data visualisation and notifications.

Therefore, it is recommended to use speech-based visualisation to alert mobile users about interesting/alarming changes and trends. Using synthesis speech instead of visual effects to alert the user produces no clutter on screen and could help the user to pay attention to the voice while handling some other tasks without looking or touching the mobile device. Having reviewed the current research trends and recommendations for audio data visualisation, we now review and analyse UDM visualisation to establish how adherent it is to the general principles of mobile data visualisation.

3.3 UDM Visualisation

Visualising the outcome of the UDM analysis is an essential task in order to enable mobile users to make use of the data analysis and to make well-informed decisions in real-time. While UDM visualisation shares the same challenges and issues of the mobile data visualisation, UDM visualisation also faces more hurdles than those that are specific to the general mobile visualisation. Therefore, in the following sections, we review the UDM visualisation challenges first, and then we identify the essential aspects that UDM visualisation systems must have. This will determine the functional requirements and capabilities of our proposed model based on what is necessary for UDM visualisation.

3.3.1 UDM Visualisation Challenges

Considering the above-mentioned limitations of mobile devices and the computational demands of the UDM visualisation, visualising the outcomes of UDM analysis on smart phones faces many challenges. To start with, the limited screen space is a key constraint that sits on the top of the UDM visualisation challenges list. According to many recent studies (Burigat and Chittaro 2005; Burigat, Chittaro et al. 2006; Gaber, Krishnaswamy et al. 2010) there is currently lack of strategies for visualising data on mobile devices. For instance, although there have been few proposed models that have succeeded in visualising the UDM outcomes on mobile devices, they still fail to address certain requirements of mobile data visualisation, which results serious clarity and accuracy concerns (Kargupta, Park et al. 2002; Kargupta, Bhargava et al. 2004; Gaber, Krishnaswamy et al 2005; Gillick, Krishnaswamy et al. 2006; Gaber, Krishnaswamy et al 2010). Moreover, given differences in the screen sizes, resolution, and number of available colours among various devices, user interaction is a key consideration for mobile visualisation since it can significantly impact the users' perception (Bertin 1981; Burigat and Chittaro 2005; Chittaro 2006).

Furthermore, when visualising UDM outcomes, it is very hard to predict the shape or the size of the results since the visualisation process is done in real-time based on the arrived data streams. Hence, interactivity, adaptivity, and scalability are key requirements in any UDM visualisation algorithm. To make it more complicated, portable devices normally have no sophisticated I/O devices. For instance, smart phones do not have a physical keyboard or mouse for flexible data entering or interacting with the device. Besides, mobile phones have no standardised way of interfacing with the user. Many platforms have their own techniques and their implementations are specific to this device or that (Paelke, et al. 2003). This complicates the UDM visualisation and to the list of its challenges.

Likewise, performance and limited power supply are also main barriers when visualising the UDM outcomes on mobile devices. In fact, until recently, UDM visualisation was not feasible on mobile phones due to their technical limitations (Gaber, Krishnaswamy et al. 2010). While smart phones normally have constrained resources, data stream mining visualisation requires reasonably powerful hardware specifications to perform such tasks as explained in chapter 2 earlier. However, by overlooking some of the mobile data visualisation general disciplines, UDM visualisation applications are still missing remarkable opportunities for improving visualisation and enhancing the resource management.

In addition, mobile data stream mining visualisation faces many other challenges. For example, Gaber, et al. (Gaber, Krishnaswamy et al. 2010) mentioned that the evolving nature of data streams requires large space of memory especially for real-time systems, where chances of storing data and dealing with it offline are very rare. This is totally contradictory to what mobile devices have of small and constrained memory systems. Consequently, for a UDM visualisation algorithm to be successful, it has to have different means of adaptivity that enables it to cope with the limited computational resources while still visualising the UDM outcomes efficiently and appropriately.

Additionally, mobility, connectivity, and adaptation to the available level of resources are also more challenges for UDM visualisation (Encarnacao, Frühauf et al. 1995; Gaber, Zaslavsky, et al. 2004; Gaber, Krishnaswamy et al. 2010). For instance, in the case of mobile UDM visualisation, every time, the visualisation process happens in different context, for different datasets, and that can notably affect the overall perception of the data visualisation (Chittaro 2006). Hence, many factors need to be taken into account when designing a UDM visualisation algorithm. Examples of such factors are auditory environment, visual environment, and the level of attention (Paelke, et al. 2003). Auditory environment refers to the auditory context, where the UDM visualisation is happening. This can be very quite or very noisy based on where the user is trying to perform the UDM visualisation. Similarly, the visual environment indicates the visual context. The location of where the UDM visualisation is happening can be very bright and shiny once, and very dark in some other times. Thirdly, mobile users are moving and handling many tasks at once. In UDM visualisation, users are expected to do many multi-tasking jobs and the probability of task interruption is also very high. Thus, this also needs to be considered as an important factor. In addition, as mobile users are moving, there is a high chance that they might lose network connection because of wether, other radiations, or any other factors. This issue, which is termed as connectivity problem, also needs to be taken into account. Overall, UDM visualisation must be dynamically adaptive not just to the level of the available resources, but also to the context of the user and his/her preferences.

Furthermore, the UDM visualisation process requires leveraging the use of all the existing communication channels between the mobile user and the device. This does not only include the visual channel (visualisation on screen), but also others such as voice and vibration (Charaniya and Lodha 2003). This is essential for facilitating the interaction between users and mobile devices in order to assure appropriate delivery of information (Paelke, Reimann, et al. 2003; Campbell and Tarasewich 2004). In fact, in their study, Charaniya and Lodha (Charaniya and Lodha 2003) mentioned that synthesis speech (voice) could be used as an effective tool for communicating to the machine. Therefore, these new means of interaction (voice, vibration, etc) could be used in mobile UDM visualisation applications in order to assure understandable information presentation and an efficient use of resources.

3.3.2 UDM Visualisation Essentials

After reviewing the mobile data visualisation and the UDM visualisation, we can now list a number of key characteristics that UDM visualisation systems must have in order to deliver comprehendible visualisation at an efficient performance and resource consumption rates. We summarise the requirements of UDM Visualisation as follows:

1. Mapping

As mentioned by Chittaro (Chittaro 2006), values and relations among results of the UDM outcomes must be mapped appropriately to different shapes and objects (lines, circles, etc) to enable users to discover knowledge and/or interesting patterns from the UDM outcomes.

2. Interaction

As explained earlier in section 3.2, interaction is a key aspect of any mobile visualisation system. In order to amplify the user's cognition by the visualised information, interaction must be supplied in UDM visualisation systems so that users can appropriately and accurately comprehend the presented data.

3. Selection ("details-on-demand")

We have learned in section 3.2 that providing too much data on the small screen space of smart phones may cause serious accuracy and clarity issues. Such cases may lead to dangerous and wrong decisions. One the other hand, providing too little information can also produce similar results, if the user was not provided with the necessary information. As a result, different means of selections ("details-on-demand") must be provided to the user to allow him/her to get more necessary information about any item at any given time. The effectiveness of selection capabilities in mobile data visualisation was demonstrated in an experimental case study by (Tonder and Wesson 2008).

4. Adaptivity

To overcome the smart phones resource constraints and the limited screen space, UDM visualisation algorithm must have a high level of adaptation (Burigat and Chittaro 2005; Tonder and Wesson 2008; Gaber, Krishnaswamy et al. 2010). However, (Chittaro 2006) mentioned that while adaptation is a key requirement for mobile data visualisation systems, it is not enough by itself for satisfying the requirements of such systems. It was claimed by (Burigat and Chittaro 2005) that adaptation of the data visualization must comply with the users' interests and needs to reflect the fast changing and evolving nature of data streams.

5. Context Consideration

Mobile phones have no particular environment that they work in (Savio, and Braiterman, 2007). Mobile users move continuously from one place to the other. This means that UDM visualisation applications will need to consider the visualisation requirements of the different context. Some of the main possible changes in the mobile user's context are; first, Auditory environment: sometimes it is noisy and others quite based on the environment and surroundings. Second, the visual environment needs to be factored in terms of the level of light. Thirdly, the level of users' attention has to be considered, and this will be different based on what the user is doing (Paelke, Reimann, and Rosenbach, 2003).

Additionally, smart phones are designed for widespread population, which means that users may have no technical background or training. Moreover, the limitations of the I/O tools, the multi-tasking nature, and the task interruption probability should all be considered as part of the context of the UDM visualisation systems (Dunlop, and Brewster, 2002).

6. Use of different communication channels

The need for using different communication channels to interact with the mobile user was explained in section 3.2 earlier. It can assure the clutter reduction through avoiding the addition of Halos or Arrows to inform the user about the off-screen objects. Speech-based visualisation can also be used to enable users to do multi-tasking when they are busy attending some other tasks in the field. It even simplifies the use of the mobile application since users are not required any more to keep continuously watching the small mobile screen, which could possibly cut the use of energy.

7. Personalise-able/User-driven Presentation

Different mobile phones have different screen size, resolution, and support different number of colours. Also, different users have different level of data perception and clutter tolerance. This might also vary based on the application, dataset, and the context of where the mobile user is. Furthermore, the outcome of the UDM visualisation is not predictable. Therefore, mobile users must be given full control to personalise, adjust, and change the UDM outcome presentation so that it can fit to their preference/application/context/and mobile device.

8. Human Factors

Basic knowledge of the differences in people perception and cognitive capabilities is needed to design better applications that can suite different people.

9. Evaluation

After the data has been visualised, the system should be tested on real users and using lab-based experiments to get feedback. We have explained earlier the two main approaches for evaluating UDM visualisation systems, field-based and lab-based. We have also explained the differences and whether it is worth the cost of implementing the field-based evaluation as indicated by Kjeldskov, et al. study (Kjeldskov, et al. 2004). Such evaluations are required to ensure the effectiveness of these systems.

UDM visualisation is in a nascent stage of evolution and the Clutter-Aware Clustering Visualiser CACV (Gaber, Krishnaswamy, et al. 2010) is a current end point for a generic UDM visualisation. We now review CACV to identify/analyse its capacity to support generic and adaptive mobile UDM visualisation in adherence to the principles of mobile data visualisation.

3.4 Review of CACV

3.4.1 CACV overview

The Clutter-Aware Clustering Visualiser is a clutter-aware and resource-aware application for real-time visualisation of mobile data stream mining (Gaber, Krishnaswamy, et al. 2010). Basically, it uses the principle of the Adaptive Clutter

Reduction (ACR). Gaber et al (Gaber, Krishnaswamy, et al. 2010) mentioned that CACV uses clutter-reduction techniques to consider the amount of information presented on the screen and dynamically adjust the data presentation so that it fits the limited screen space of the portable device. CACV focuses particularly on point-based clustering algorithms, where typical algorithms include Light-Weight Clustering (LWC) (Gaber, et al 2005), Resource-Aware Cluster (RA-Cluster) (Gaber and Yu 2006), and Very-Fast K-Means (VFKM) (Domingos and Hulten 2001). In the context of UDM, Point-based Clustering techniques process the incoming data streams and assign the new data items – in real-time – to existing clusters or create new clusters based on distance measures (Gaber, Krishnaswamy, et al. 2010). Also, CACV performs adaptation of the clustering process according to the available resources, incoming data rates and rate of generating new clusters.

In CACV, clutter is measured in terms of two diverse attributes in correspondence to how clutter is defined in the literature in terms of the amount of information presented on the screen. The first attribute is the one that pertains to the percentage of the screen occupied by the clusters generated and shown on the screen. The second one pertains to the percentage of clusters that overlap/intersect with one another on the screen. If this amount of clutter or some other aspects of the presented information has the potential of reducing the understanding/comprehension of the presented information, then this can be treated as clutter (Liono 2009; Gaber, Krishnaswamy, et al. 2010).

As explained earlier, on one hand, mobile devices generally have small screen space. On the other hand, UDM visualisation requires analysing massive and continuous data streams continuously and presenting the outcomes to the user. Logically, traditional approaches for visualising such massive and continuous streams on small screen will certainly produce too much clutter. In order to reduce this amount of clutter, CACV implements four modes of UDM visualisation (Gaber, Krishnaswamy, et al. 2010), as follows:

1. Normal Mode

This is the start up mode and it refers to analysing the incoming data items using the LWC algorithm, then drawing clusters on the screen. Each cluster size reflects the number of points/data items in that cluster. As the data streams keep on coming, the size and the number of clusters increase notably as shown in figure 3.1. Once clusters are drawn on the screen, a periodic assessment will be implied to examine the amount of clutter on screen.

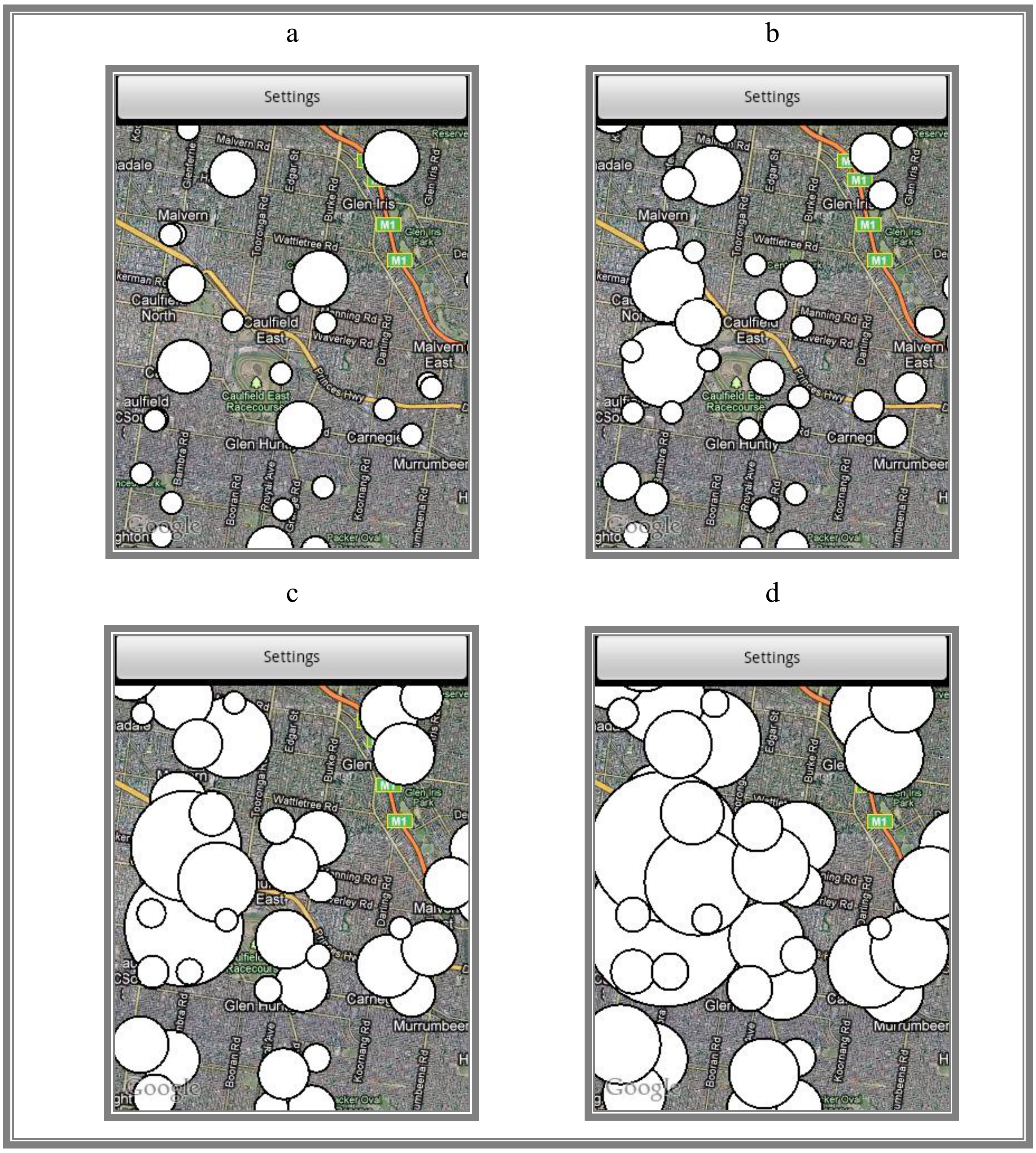


Figure 3.0.1 shows the normal visualisation mode in CACV

2. Scaling Mode

Once a certain predefined/static value (Coverage Threshold) is reached, the visualisation will be switched into the scaling mode. In Scaling mode, CACV attempts to reduce clutter by reducing the size of all clusters by a predefined *ScaleFactor* under the condition that the smallest cluster size is not less than a cluster with only one point in the normal mode. As the number and the size of clusters grow, CACV scales all the clusters by the $ScaleFactor$. Likewise, Based on the periodic assessment of the coverage and the overlap of clusters, if the coverage is still higher than the predefined threshold, CACV will switch to the colouring mode.

3. Colouring Mode

Colouring modes is another attempt of CACV to reduce clutter using Colour Changing, which is another clutter reduction technique. In Colouring mode, CACV aims to show distinctive visual appearance of clusters using colours. In other words, in colouring mode, CACV uses colours to indicate the weight of each cluster while maintaining the on-screen size of clusters (circles) the same for all clusters, at the minimum size.

The density of the colour refers to the number of items in each cluster. This means that the darker the cluster is, the larger number of data items in that cluster. This technique is very helpful for users to differentiate clusters and comprehend the outcome of the data analysis while keeping the level of clutter at lower level as shown in figure 3.2.

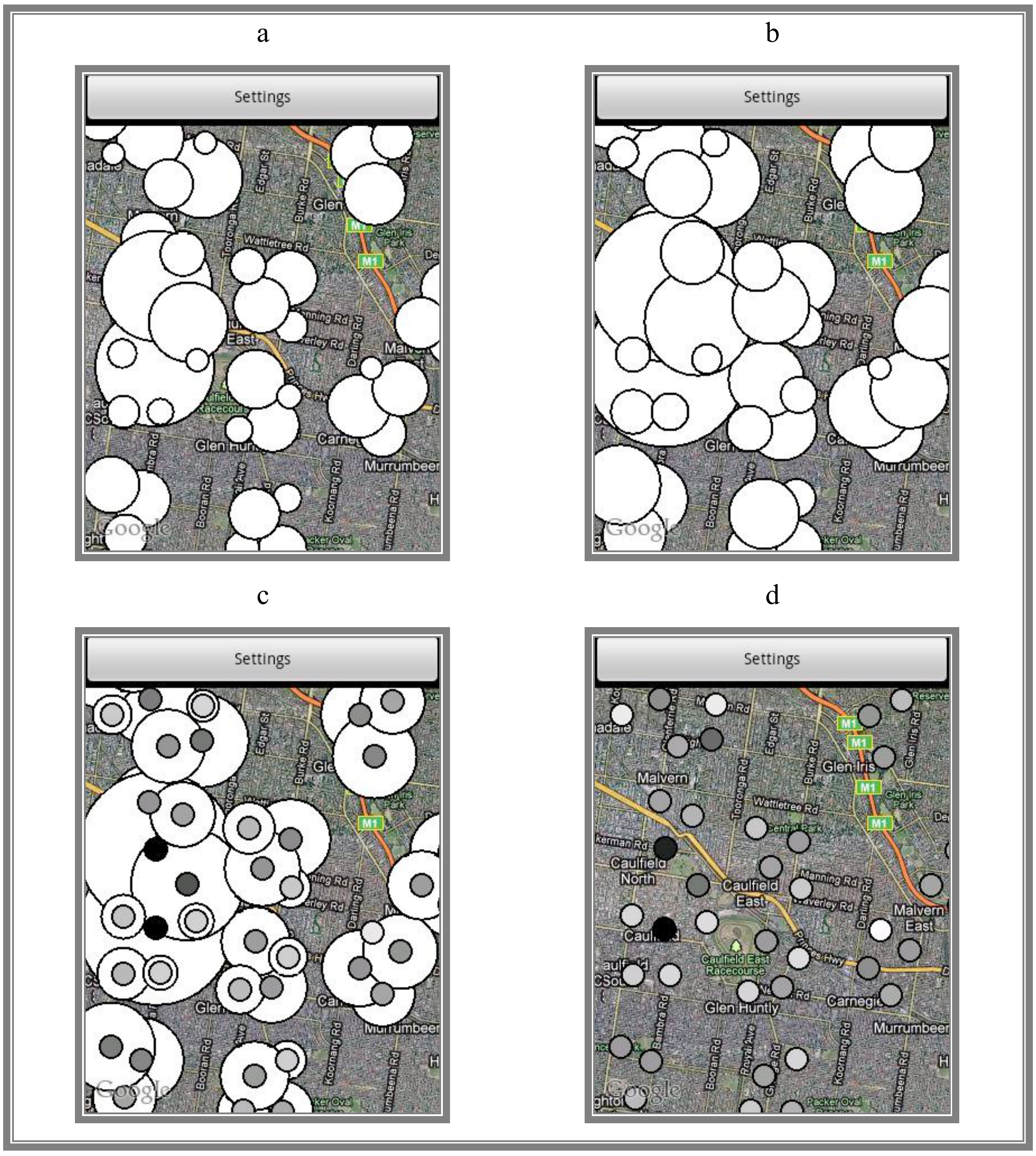


Figure 3.0.2 show the colouring visualisation mode in CACV

4. Sampling Mode

This refers to selecting only active clusters to be presented on the screen. Active clusters, in this context, means the ones that have attracted new data points in the most recent time intervals. This visualisation mode is used by CACV as the last mode of adaptation, in case some of the aspects of clutter are still higher than the predefined threshold after switching to the colouring mode. The aim of using this mode is also to reduce clutter and to shift the focus of users to the active clusters. It is worth mentioning here however that switching from one visualisation mode to the other is done totally by the CACV algorithm, which is not always appropriate as it is be based on predefined parameters.

3.4.2 CACV Limitations

From figure 3.1 - d, it can be seen that the there is a serious clarity issue with the data visualisation. That is because users are not able to differentiate clusters, not to know how many data items in each one, neither what is the region of each cluster or where they are located. Addressing such concerns is the main aim of mobile data stream mining visualisation in order to enable users to make better and well-informed decisions in time critical scenarios. However, delaying the switching from the scaling mode (high clutter) to the colouring mode (less clutter) can result serious concerns about users comprehension and/or possibly misinterpretation of the visualised data.

Furthermore, although changing the values of different parameters such as coverage and overlapping thresholds can reduce clutter and improve efficiency, CACV does not offer any means of interaction to change any of those values. Enabling the user to change these values dynamically (during run-time) is vitally important to avoid any data visualisation accuracy and/or clarity issues. That is because different mobile phones have different screen size, different resolution, and supports different number of colours. Also, different users have different level of data perception and clutter tolerance based on their personality, device, and application.

Another performance issue of CACV system is that it visualises all incoming data objects regardless of whether they are on or off-screen, which in turn can reduce performance and efficiency. Such inefficiencies can be avoided by incorporating different means of selection and interaction. It is also worth mentioning that CACV algorithm currently provides no alert or visualisation for the off-screen objects, which could lead to wrong and dangerous decisions if the mobile user was not aware of the overall image of the UDM results.

3.5 Conclusions

In summary, this chapter has reviewed mobile data visualisation generally and the UDM visualisation in particular. We have identified the challenges, trends, and techniques of UDM visualisation. We have also developed a list of the essential characteristics of typical UDM visualisation systems.

According to our literature review, there are currently no generic strategies or guidelines for mobile data stream mining visualisation. In the state-of-the-art project (CACV) that we have reviewed, many essential visualisation requirements were missing such as interaction, selection, and user-driven data presentation. The absence of such capabilities might cause the data visualisation to be ambiguous, which could lead to false information delivery and wrong decisions. This ignorance of interaction and other visualisation requirements can affect not only the data visualisation process but also the performance of the whole model as in the case of CACV when it visualises all the incoming data items regardless of their positions. This could cause the performance to drop notably and to also drain the battery quickly.

There are number of reasons why these interaction capabilities are important and essential for UDM visualisation systems. To start with, mobile devices have different screen sizes, different resolutions, and they support different number of colours. Therefore, assuming a particular static threshold or colour as in the case of CACV may work for some devices and/or users but not for all. This could cause a serious issue if for instance the colour system does not support all the colours that are representing the clusters, which could result in misleading data visualisation.

Additionally, mobile users have different level of data perception and clutter-tolerance. For some people if the screen was occupied by more than 30%, the screen is cluttered, while others might tolerate even up to 60% of cluttered screen. This can be seen as a result of different personality, variation of datasets, and also can be affected by the type of application.

Furthermore, in UDM visualisation, the main goal is to visualise the outcome of analysing a continuous and rapidly changing flow of data items. This makes it almost impossible to predict the shape of the output or the number of clusters that might be produced since it is a continuous stream of data. Yet to make it harder, this continuous flow needs to be analysed and visualised in real-time with no chance of storing it for off-line analysis.

Finally, in almost all of the reviewed applications in UDM visualization area, the limited screen space was the only media of communication between the mobile user and the machine. This causes the screen to be cluttered quickly because of the rapidly increasing number of incoming data items. As shown in our review, other existing communication channels such as synthesis speech and vibration can be utilised to provide efficient, elegant, and clutter-free feedback to the mobile user. Currently, these types of HCI (Human Computer Interaction) communication channels (voice and vibration) are not used in the UDM visualisation area.

To conclude, there is an increasing demand in the field of mobile data stream mining visualisation for an innovative, generic, adaptable, and interactive visualisation model that can provide accurate data visualisation functionality at a reasonably acceptable level of performance. New communication channels are required to ensure accurate delivery of the presented information while maintaining the least level of clutter on screen.

4 iCACV: Interactive Clutter-Aware Clustering Visualiser

4.1 Introduction

Visualising the outcome of the mobile/ubiquitous data stream mining is crucial for mobile users who need to be well informed in order to make decisions in real-time. As explained earlier in chapter three, data visualisation gives the user a better chance to understand and appropriately comprehend the feedback message that is provided by machine(s). However, UDM visualisation, as an emerging field, faces many serious challenges such as the limited screen space, limited energy supply, and constrained resources.

In chapter 2, we have reviewed the few available UDM visualisation systems such as VEDAS, and CACV (Kargupta, Bhargava et al. 2004; Gaber, Krishnaswamy, et al. 2010). While these systems represent the state-of-the-art in this field, UDM visualisation is still very far from its prime as it is still lacking of any generic and adaptive algorithm that can be used for analysing and visualising any dataset. Chittaro (Chittaro 2006) also pointed out that mobile data visualisation in general is still evolving and professional practices and successful-proven guidelines are still missing in this context.

Gaber et al (Gaber, Krishnaswamy, et al. 2010) reasoned the absence of a generic visualisation algorithm to the fact that it is only now that we are able to perform such operations on mobile devices due to the unprecedented and remarkable growth in portable devices capabilities. However, as now we have such UDM visualisation applications that are leveraged to operate on smart phones, the focus should be shifted to improving such systems to ensure their effective and practical operation. In this context, addressing the following issues are the main concerns in the UDM visualisation today. First, maintaining the data visualisation accuracy and clarity so that to ensure that users are given the appropriate information that is necessary for making decisions. Secondly, reducing the energy consumption, which is crucial for guaranteeing the operation of those algorithms for reasonably acceptable amount of time. Having an algorithm that can perform the visualisation on mobile phones but drains the battery in few minutes will not be of help for mobile users. Thirdly, in order for a UDM visualisation algorithm to be successful, it needs to have a continuous adaptation to the available resources such as memory, and processor.

Addressing the above-mentioned concerns of UDM visualisation as well as issues like mobility and connectivity, is a necessity for providing an interactive, generic, and adaptive UDM visualisation application.

Hence, this thesis aims to develop an interactive, adaptive, and user-driven UDM visualisation strategy based on the CACV algorithm to enhance the data visualisation, and the resources management. Since we now have a list of the UDM visualisation essentials, we attempt to address the missing aspects in the present UDM Visualisation systems to achieve the objectives of this thesis.

This chapter is structured as follows:

- Section 4.2 presents our proposed model that incorporates all our proposed capabilities for interactive UDM visualisation. This model is based on our list of the UDM visualisation essentials as in chapter 3.
- Section 4.3 presents the algorithm and the theory of our proposed model iCACV.
- Section 4.4 presents the iCACV framework.
- Finally, section 4.5 summarises this chapter and presents the conclusions.

4.2 iCACV: Interactive Clutter Aware Clustering Visualiser

As explained earlier, our proposed model is based on CACV that was presented by Gaber et al (Gaber, Krishnaswamy, et al. 2010). We propose incorporating different means of interactivity to improve different aspects of the UDM visualisations in CACV. With our enhancements, we aim to address the concerns that were listed earlier, which were overlooked in CACV. The objective of this model is to reduce clutter, improve performance, and reduce energy usage. The iCACV involves incorporating the following features discussed in this section.

4.2.1 Dynamic Setting of Visualisation Thresholds

According to Spence (Spence 2001), mobile users must be in continuous control over the data visualisation process to perceive and understand the visualised data. This is because different portable devices have different screen size and different resolution. Also, the users themselves have different level of data perception. Therefore, we propose developing user-driven data visualisation mechanisms. This means enabling users to control what is displayed on screen. In other words, we provide users with the power to dynamically personalise and adjust the visualisation parameters based on their preferences. For instance, figure 4.1 shows a screenshot from the CACV system, where it was assumed that the algorithm would switch from the normal mode to the colouring mode at a static coverage threshold of 60%. However, looking at picture 4, it can be easily seen that the user will not be able to identify the different clusters due to their overlap although the specific clutter/overlap thresholds have not been reached. This is a clear limitation in static and fixed setting of these thresholds. To address this, we propose providing visual controls for the user to change various algorithm parameters such as coverage and overlapping thresholds at run-time. This way, the algorithm can be personalised to individual user/device/analysis needs.

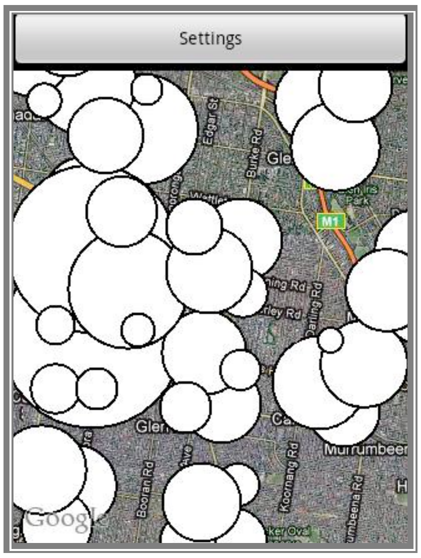


Figure 4.0.1 shows clusters in the normal visualisation mode in CACV

CACV, as explained earlier, uses clutter reduction techniques. Generally speaking, those techniques or approaches, as explained earlier, are implemented based on certain parameters or thresholds. Some of those parameters are important in terms of controlling the level of clutter on screen and/or adapting to the available resources level. We list those parameters, which we propose to make personalised, to enable users to control, adjust, and better comprehend the on screen information:

- **Coverage Threshold (*CT*)**

CACV algorithm uses this value, which ranges from 0.1 to 1, to decide whether to switch to another visualisation mode or not. This is done by comparing the value of the on-screen coverage percentage to this value. Basically, 1, which is the highest value of *CT*, means that the algorithm will allow coverage of up to 100% of the available screen space. This means that setting the *CT* to 1 will result in no clutter reduction. Oppositely, setting *CT* to 0.1 will enable the algorithm to switch to another visualisation mode to reduce clutter once the on screen clutter exceeds 10% of the total screen space.

- **Overlapping Threshold (OT)**

This refers to the maximum portion of a cluster that is allowed to be overlapped/interfered with other clusters. It also has a value range of 0.1 to 1, to control the visual overlap of clusters. Setting the OT to 1 enables clusters to totally overlap/interfere with each other. However, setting it to 0.1 will allow clusters to overlap with no more than 10% of the size of that cluster. If a cluster were overlapping with another one by more than 10%, the algorithm would then switch to another visualisation mode to reduce this clutter and to help the user to differentiate clusters on screen.

- **Maximum Number of Clusters (MNC)**

Given that UDM involves analysing a continuous and "potentially infinite" stream of data items, visualising the UDM outcomes may produce a huge number of clusters on the screen. This cannot be afforded on the very limited screen space of the mobile phone. Hence, CACV performs clutter reduction by limiting the number of clusters that can be drawn on the screen based on this parameter. The value of the MNC ranges from

1 to max, where max equals to the total number of clusters that can be displayed at any point of time on the screen.

4.2.2 Selective Focusing

It can be seen from figure 4.1 that clusters can be totally overlapped. Some of them may lay over the others. This raises issues of data visualisation accuracy and clarity since the main aim of data visualisation is to help users in making better and well-informed decisions. For instance, users would not know which cluster in which area, and neither would they know what is the weight of each cluster. Therefore, we propose using a key character of data visualisation systems, namely selective focusing. Selective focusing refers to the process of enabling interaction (touching/tabbing) to provide detailed information about clusters. It is also called "detail-on-demand" as in our literature review in chapter 3. This ensures that brief information that is necessary to make sense of the data is provided to the user as well as the full content is also available for the user on his/her request (touch/tab/click). Charaniya and Lodha (Charaniya and Lodha, 2003) have already shown the effectiveness of this technique in their study.

4.2.3 Controlling the Clusters Growth

Usually, clustering and visualising large datasets result in a number of clusters that vary in sizes. While few of those clusters grow at a rapid rate, few others stay at a very small size. It is also possible that many clusters grow to medium certain size and stay at that size for a long time. In typical mobile data mining applications, the large clusters are the significant ones to the user since they give an alert that something needs attention in that area. In order to more optimally control the screen clutter levels, we propose to dynamically decide when clusters should start growing by specifying a cluster growth threshold. This eliminates the clutter that is generated by the delta changes of clusters. We also indicate the weight of each cluster while they are less than the cluster growth threshold. This mechanism can be accomplished by checking the cluster's weight against the cluster growth threshold. Users should be able personalise the cluster growth threshold to decide which cluster to grow and at which value.

4.2.4 Audio Feedback for Off-screen objects

There have been many studies about visualising off-screen objects to give an accurate feedback to the user while maintaining the least level of clutter on screen (Burigat, et al. 2006; Gustafson and Baudisch 2008). Some of these studies propose placing signs or pointers to indicate the locations of the off-screen objects and alert the user (Burigat, Chittaro, et al. 2006; Gustafson and Baudisch, 2008). Examples of such signs are arcs (Halo), arrows (Arrows), and lines (CityLight) as listed by Burigat et al. (Burigat, et al. 2006). However, it can be noticed that such additions of information produces additional clutter on the screen itself by adding more signs or pointers. Moreover, these techniques are error prone since their operation is based on having a sign that refers approximately to the exact location. Thus, the user is not guaranteed to be given the exact appropriate information about the off-screen objects.

In most new smart phones, text-to-speech technology can be used to enable users to gain feedback and insight of the off-screen clusters without increasing the clutter on screen. We argue that implementing interactivity via audio feedback is not only an effective manner of communicating with the user, but it can also be especially useful for mobile users where they cannot keep looking at the screen at all times and may need a hands-free type of visualisation. Likewise, when the battery level goes down, users cannot keep the screen ON for long period of time. In this context, audio feedback plays in a critically important role to enable users to gain feedback without a need to switch their mobile screen ON. Incorporating such attributes to the algorithm should provide audio alarms whenever clusters grow to a user-defined audio threshold.

4.2.5 Screen Fencing of Clusters

While the CACV offers an innovative adaptation of clusters for clutter-reduction purposes, it does the adaptation for all the clusters, regardless of whether they are on or off-screen. This consumes additional computational and energy resources. Therefore, we propose making the system aware of the dimensions of the on-screen (visible) coordinates. Thereby, before any adaptation or visualisation is done, the new incoming data item is checked, if it lies within the on-screen map dimensions, the object will be clustered and visualised. Otherwise, the new object will only be clustered in the backend without visualisation. This can improve the overall computational elegance and effectiveness of iCACV by using less computational resources and consuming less energy.

4.3 iCACV Algorithm

In order to better understand the iCACV algorithm, we first present the original CACV algorithm as it was presented by Gaber et al. (Gaber, Krishnaswamy, et al. 2010). This is shown in algorithm 4.1. It can be seen that CACV sets the visualisation parameters at the beginning of the algorithm and use these values to evaluate the on-screen clutter with no means of interacting with the user. As explained earlier, such interaction is essentially important to enable the user to understand what is displayed on the screen based on his/her context, mobile phone, application, and the other factors that might affect the data visualisation. Algorithm 1 shows the different visualisation modes in CACV and the switching between these modes that is based on the static/predefined visualisation parameters.

Algorithm 4.1: CACV Algorithm

```
/*
 *  Clutter Aware Clustering Visualiser (CACV) Algorithm
 *  Copyright 2010 Monash University. All rights reserved.
 */

10      {User Settings}
20      input CT, OT, NocThreshold, RefSc
30      input CGEnabled, CGThreshold, AudioEnabled, AudThreshold
40      repeat
50           if RefSc is reached then
60                 {Draw Clusters}
70
80                 for each cluster ci  do
90                       Draw according SET (V IS)
100                      SET(V IS) = Area(ci)αwi
110                end for
120
130                Calculate ScCov, OvClu
140                {Adaptation Level 1: Scaling}
150                if ScCov > CT v OvClu > OT then
160                      ScaleFlag ← False
170                      for each cluster ci  do
180                            wi′ ←wi × ScaleFactor
190                            if wi′ < 1 then
200                                  ScaleFlag ← True
210                                  Break
220                            end if
230                      end for
240
250                      if StateFlag = False then
260                            SET(V IS) = Area(ci)αwi′
270                            Calculate ScCov, OvClu
280                      end if
290                end if
300
310                {Adaptation Level 2: Shading}
320                if ScCov > CTvO v Clu > OT then
330                      SET(V IS) = Color(ci)αwi
340                end if
350                Calculate ScCov, OvClu
360                {Adaptation Level 3: Cluster Selection}
370                if ScCov > CTvOvClu > OT v NumOfClusters >
NocThreshold then
380                      SET(V IS) = Color(active(ci))αwi
390                end if
400          end if
410     until END OF THE STREAM
```

Algorithm 4.2 shows the incorporation of our proposed enhancements that can be incorporated into the CACV algorithm in order to provide an interactive, generic and user-driven UDM visualisation. The lines (7 – 26) represent the interactive part of the algorithm. Like CACV, iCACV sets the visualisation parameters when starting the data analysis. However, consistent monitoring has been implemented to respond to any change that the user might desire.

Line 90 of algorithm 4.2 shows the Clusters' Screen-Fencing technique that aims to reduce the computational and energy resources by not visualising the off-screen objects. If a data item is off-screen, then the Audio Feedback feature will be checked. If the user has enabled the Audio Feedback, a meaningful audio message will then be played to the user once a cluster exceeds a particular weight threshold. This can be seen in line 180 of algorithm 4.2. A similar approach can be enabled for on-screen objects, where users, who have some other tasks in hands, want to get instant alert about any alarmingly growing cluster without the need to keep looking at the screen.

When drawing clusters, iCACV algorithm checks whether the Cluster Growth Control is enabled. If so, all clusters will be drawn as circle with the minimum size, unless a cluster exceeds a certain threshold (Cluster Growth Control Threshold CG). When using Cluster Growth Control technique, clusters' weight is shown as numerical value on the top of each cluster. Once a cluster attracts a number of points that is greater than the CG threshold, cluster's weight is then reflected in the size of the circle as in lines 110 to 150 in algorithm 4.2.

Moreover, to guarantee easy to comprehend graphs, users in iCACV are given the control to personalise the data visualisation by changing the visualisation parameters according to their context and preferences. This can be seen in lines 230 to 250, where users are capable of changing the important visualisation parameters at any point of time to ensure user-centric visualisation. Also, the last type of interaction was implemented as Selective Focusing, where users are given the power to obtain detail information about any cluster at any point of time to ease comprehension and interpretation of the data visualisation. This is shown in lines 270 to 290.

Algorithm 4.2 iCACV algorithm

```
/*
 *
 *  Created by Hasn AlTaiar on 22/10/10.
 *  Copyright 2010 Monash University. All rights reserved.
 */

10     {User Settings}
20     input CT, OT, NocThreshold, RefSc
30     input CGEnabled, CGThreshold, AudioEnabled, AudThreshold
40     repeat
50          if RefSc is reached then
60                {Draw Clusters}
70                for each cluster ci  do
80                {Interactive clutter-Aware clustering}
90                      if ( ci IS On-Screen ) then
100                           Draw according SET (V IS)
110                           if ( NumEnabled  ^  Area(ci) < NumThreshold) then
120                                 SET(V IS) = Minimum Area (ci)
130                           else
140                                 SET(V IS) = Area(ci)αwi
150                           end if
160                     else
170                           if (AudioEnabled) then
180                                 AudioAlarm(Speak ci)
190                           end if
200                     end if
210               end for
220
230               if (CT ∨ OT ∨ NocThreshold ∨ NumThreshold ∨ AudThreshold IS Changed) then
240                     Update values (CT, OT, NocThreshold, NumThreshold, AudThreshold)
250               end if
260
270               if (any Cluster IS Chosen) then
280                     Display (Cluster Detail) ∨ Speack (Cluster Detail)
290               end if
300               {End of the interactive part}
310
320               Calculate ScCov, OvClu
330               {Adaptation Level 1: Scaling}
340               if ScCov > CT ∨ OvClu > OT then
350
360    {The rest is the same as the CACV Algorithm}
370
380    end if
390   until END OF THE STREAM
```

4.4 iCACV Framework

The iCACV framework is basically developed on top of the CACV model, and thus it supports the resource-awareness and the energy-awareness since they are already incorporated in CACV. Our premise is that the CACV algorithm can be improved by using different means of interaction that can enhance performance, reduce clutter, and improve the resources management. Figure 4.3 shows the basic block diagram of iCACV Framework. Our user-driven approach of data visualisation represents another layer of clutter-awareness, which can be placed on top of the resource awareness and the energy awareness.

The user-driven data visualisation layer that is on top of the CACV framework consists of different modules that are responsible for addressing the UDM visualisation challenges through interaction. Firstly, the Cluster Growth Controller module is in charge of controlling the growth of clusters on screen. In other words, if the Cluster Growth technique were enabled, the Cluster Growth Controller would not allow clusters to grow on the screen until they reach into a certain dynamically controlled threshold (cluster growth threshold).

Similarly, the Audio Feedback handler component is responsible for managing the audio alerts when the audio feedback feature is enabled. This module examines all clusters and once a cluster grows beyond a certain threshold (Audio Threshold), an audio alert is given to the user to bring his/her attention to the current trend.

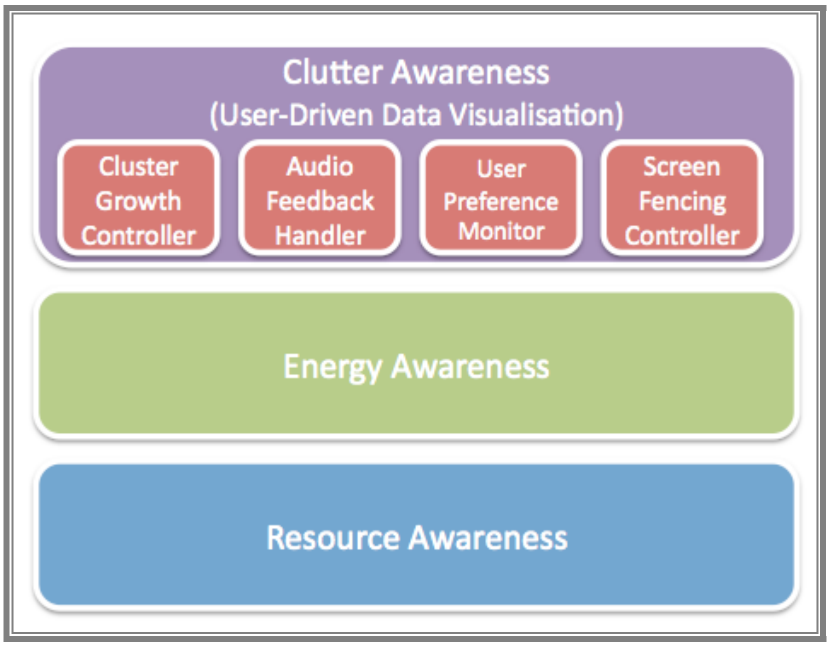


Figure 4.2 shows iCACV Framework

Thirdly, the User Preference Monitor is responsible for monitoring any preference that the user may want to change to personalise or update the data visualisation. It continuously checks for any input from the user. Once any input is received, it takes the appropriate action in updating the visualisation parameters and reflects the changes on the data presentation on screen. Another task of the User Preference Monitor is to respond in case the user requested detail information about a particular cluster and ensure the user's comprehension of the presented data.

Finally, the Screen Fencing Controller is responsible for assuring that only on-screen objects are visualised in order to reduce any unnecessary computational overhead. It also checks when a cluster grows beyond a user-defined threshold and sends a message to the Audio Feedback Handler to alert the user about that off-screen object.

4.5 Conclusions

In this chapter, we have presented our proposed model iCACV as an interactive clutter aware clustering visualiser. It aims to address the UDM visualisation essentials by enhancing data visualisation and improving resource management in CACV by incorporating different means of interaction, selection, and preference-based data presentation. We have shown an effective approach to address the main issues of UDM visualisation in order to provide generic, adaptive, and interactive UDM visualisation strategy, which is the goal of this thesis. The algorithms and the theory of both CACV and iCACV were also presented in this chapter. Finally, we have demonstrated an abstract multi-layer representation of the iCACV model to explain the main components of the framework and to facilitate the process of implementing it.

5 Implementation and Evaluation

5.1 Introduction

This chapter presents the implementation of our proposed model (iCACV) as well as the mechanisms to evaluate its effectiveness. To start with, the architecture of iCACV is presented to demonstrate our implementation of iCACV. The second part of this chapter focuses on the evaluation of iCACV. The experimental setup, datasets, and the system architecture are presented. We discuss the results of our experimental test at the end of this chapter to examine the effectiveness of the user-driven UDM visualisation model (iCACV).

5.2 Implementation

5.2.1 Overview

As we have presented the algorithm and the framework of iCACV, this section focuses on presenting the implementation and the development of iCACV model to provide a generic and adaptive UDM visualisation strategy. Our development is based on Google Android mobile phone that was designed for developers since it gives powerful capabilities for developers to extensively utilise the available features of the device. As part of the implementation and evaluation, we present the technologies that we have used. As for the implementation, we used Google Android phone[4] and Eclipse[5] as software development environment. Thus, we review both of these systems in the next section.

[4] Details of Google Android Dev Phone 1 can be found in http://developer.android.com/guide/developing/device.html#dev-phone-1
[5] Details on Eclipse software development environment IDE can be found on http://http://www.eclipse.org/

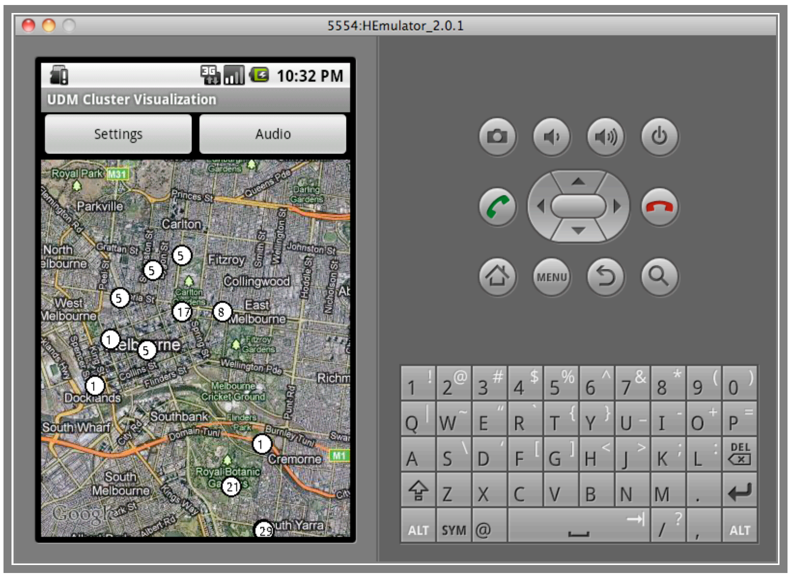


Figure 5.1 shows basic implementation of iCACV on Google Android Emulator

Figure 5.1 shows a screenshot of the basic implementation of the iCACV model that was developed based on the CACV algorithm. A comprehensive description of the scenario and the data that is analysed and visualised on the mobile phone is provided in the evaluation section later. When developing and testing iCACV, both real Google Android mobile phone and the emulator that comes with the Android SDK were used for testing and evaluating the model (iCACV).

An operational version of the CACV is already available as it was developed by (Gaber, Krishnaswamy et al. 2010). This is very useful for the evaluation to compare and contrast the results of testing the two models to examine any possible improvement. It also provides a very rich underlying framework to develop the iCACV based on.

5.2.2 Technologies in Use

- **Mobile Phone**

Google Android Dev Mobile Phone 1[6] is the phone that is used for all the implementation and evaluation in this thesis. One of the reasons for using this device is that it operates on Android Operation System (OS), which is developed by Google Inc

[6] Details of Google Android Dev Phone 1 can be found in http://developer.android.com/guide/developing/device.html#dev-phone-1

and the Open Handset Alliance (OHA) under the General Public License (GNU). Android OS is developed on top of Linux[7] kernel, which is also a free product that is released under the General Public License (GNU). This means that software developers can use it for developing their applications for free under the General Public License (GNU) v2. Also, Android OS supports developing applications using the very well known Java language, and it supports most of the libraries the comes with Java Development Kit (JDK). In Android, this is done by incorporating its own Java Virtual Machine (JVM) that makes the Android mobile phone capable of supporting Java applications just as good as the normal Java Standard Edition (J2SE). Previously, most mobile phones support Java Micro Edition (J2ME), which is basically a minimum version of java with limited capabilities to enable its operation on mobile devices. Hence, Android OS offers a very attractive platform for mobile software developers to make use of the available full support of Java language to produce interactive and powerful applications. Furthermore, being an open source product, Android OS attracts more developers and grapes more attention since it is free to use and developers can read and modify the source code of the operating system, which gives developers more power in developing their applications.

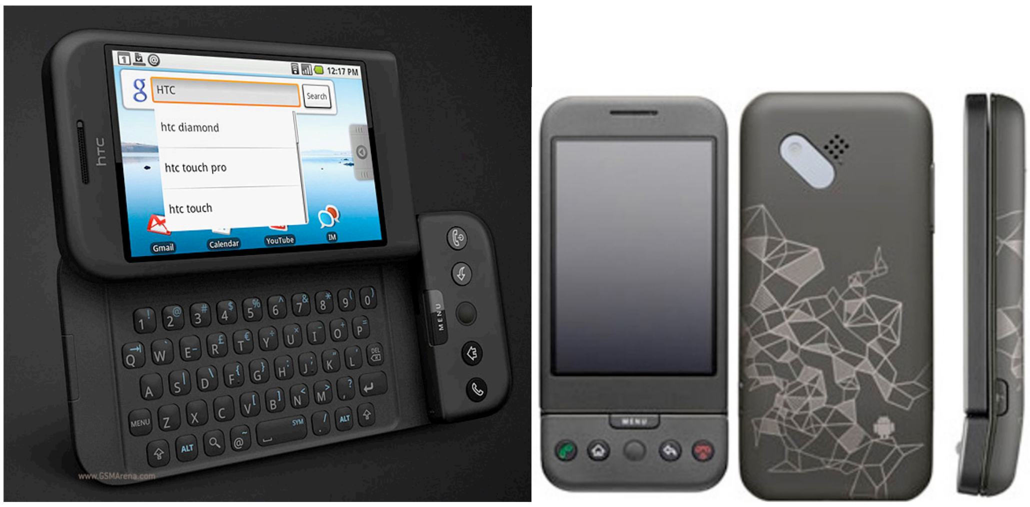


Figure 5.2 HTC Dream (left) and Google Android Dev Phone 1 (right) (GSM Arena and PhoneArea, 2009)

[7] General Public License of Linux kernel http://www.linux.org/info/gnu.html

The Google Android Dev Phone 1 is basically designed for developers to debug and develop applications for Android. A similar device was assembled for consumers' use, which is named HTC Dream phone, as shown in figure 5.2 (left). One of the advantages of having a developer phone is that it comes in an unlocked mode, which means that developers have the freedom to fully utilise the phones capabilities. It also allows developers to change the OS version and the SDK through "Flashing" technique. Such task of changing the phone's firmware is not available in the consumers' version of Google' mobile phone.

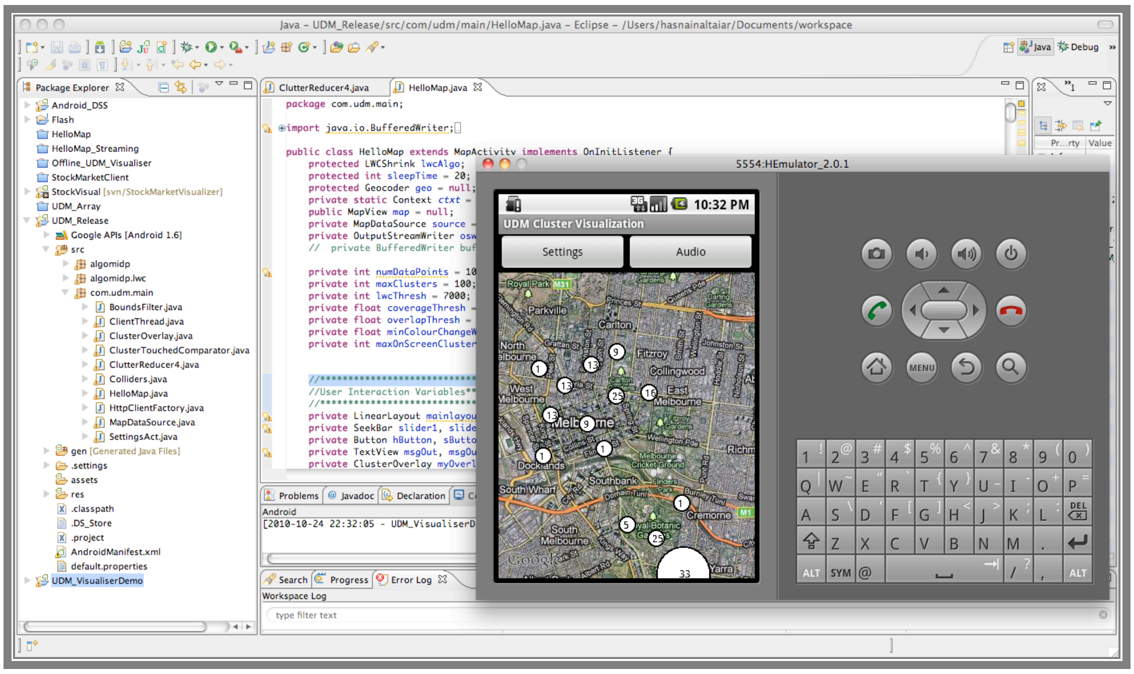
In brief, using Google Android Dev Phone 1 enables us to utilise the full capabilities of the phone, while making sure that the phone is capable of performing the UDM visualisation, given its powerful resources. The technical specification of the Google Android Dev Phone is shown in Table 5.1.

Processor	Qualcomm® MSM7201A™, 528 MHz
Operation System	Android™
Memory	ROM: 256 MB RAM: 192 MB
Network	HSPA/WCDMA: - 2100 MHz - Up to 7.2 Mbps down-link (HSDPA) and 2 Mbps up-link (HSUPA) speeds Quad-band GSM/GPRS/EDGE: - 850/900/1800/1900 MHz
Connectivity	Bluetooth 2.0 Wi-Fi®: IEEE802.11 b/g
Battery	Rechargeable Lithium-ion battery Capacity: 1150 mAh
AC Adapter	Voltage range/ frequency 100 ~ 240V AC, 50,60 Hz DC output: 5V and 1A

Table 5.1 Android Dev phone 1 technical specifications (HTC Corp, 2009)

- **Eclipse IDE**

For developing iCACV framework, Eclipse platform was used as a software development environment (IDE) since it provides good support for Android applications development through a simple add-on that is called ADT[8] (Android Development Tools). It is also free for use and released under the Eclipse Foundation Software User Agreement[9], which makes it easier to develop applications and to change the code of the core platform. The Eclipse development platform is shown in figure 5.3 during the development of iCACV framework.



```
package com.udm.main;

import java.io.BufferedWriter;

public class HelloMap extends MapActivity implements OnInitListener {
    protected LWCShrink lwcAlgo;
    protected int sleepTime = 20;
    protected Geocoder geo = null;
    private static Context ctxt =
    public MapView map = null;
    private MapDataSource source =
    private OutputStreamWriter osw
    //  private BufferedWriter buf

    private int numDataPoints = 10
    private int maxClusters = 100;
    private int lwcThresh = 7000;
    private float coverageThresh =
    private float overlapThresh =
    private float minColourChangeW
    private int maxOnScreenCluster


    //*****************************
    //User Interaction Variables**
    //*****************************
    private LinearLayout mainlayou
    private SeekBar slider1, slide
    private Button hButton, sButto
    private TextView msgOut, msgOu
    private ClusterOverlay myOverl
```

```
Android
[2010-10-24 22:32:05 - UDM_VisualiserD
```

Figure 5.3 The software development environment for developing iCACV (Eclipse) and the Android Emulator

5.2.3 iCACV System Implementation

The iCACV is a powerful framework that aims to offer a generic UDM visualisation strategy that combines the strengths of the CACV (context awareness and energy awareness) and the adaptive clutter reduction techniques. It also attempts to enhance

[8] Android Development Tool (ADT) as presented by Google http://developer.android.com/guide/developing/eclipse-adt.html
[9] Details of Eclipse software license can be found at: http://www.eclipse.org/legal/termsofuse.php

the data visualisation by making it more interactive and generic. Figure 5.4 shows the data flow in iCACV system.

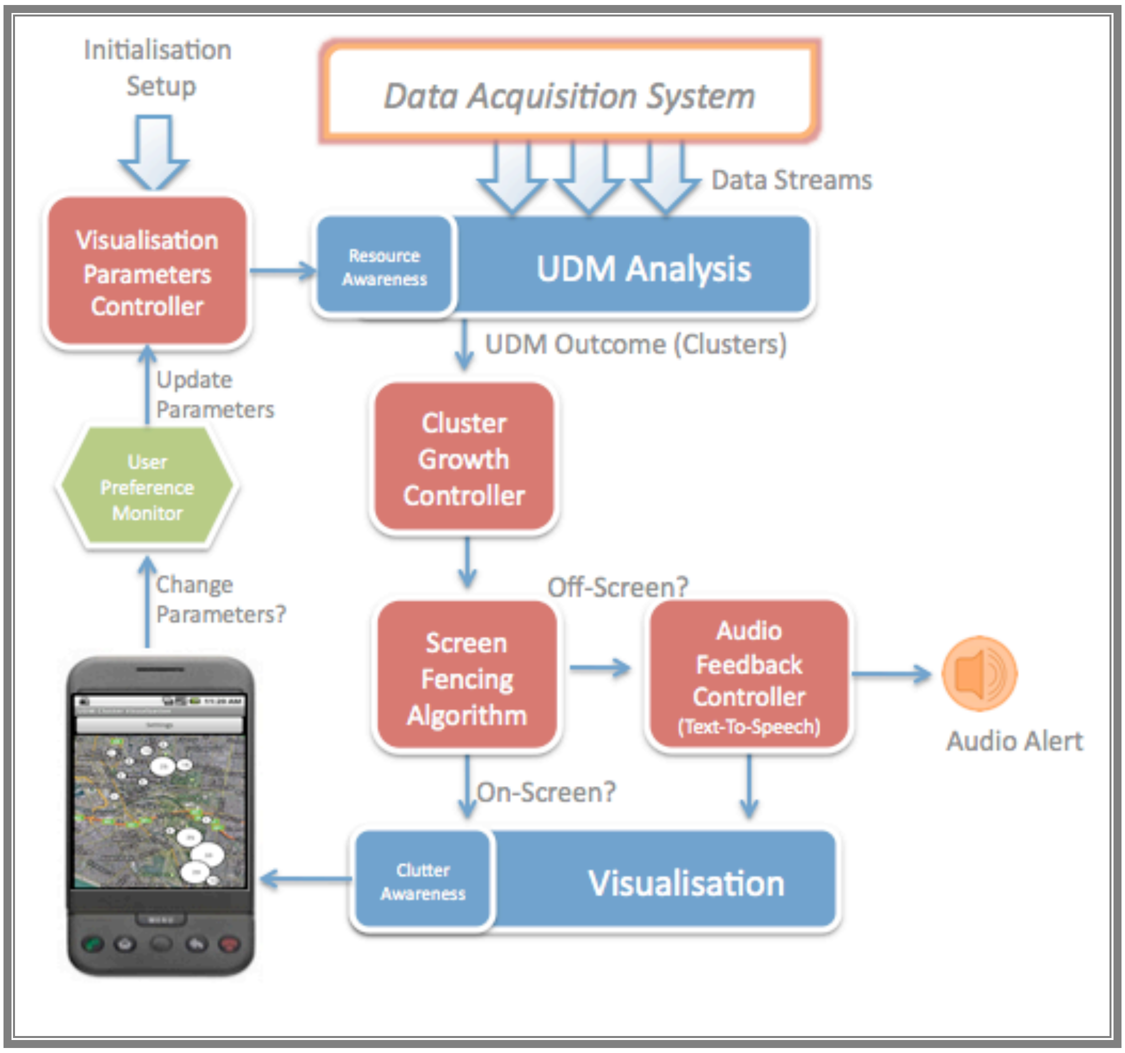
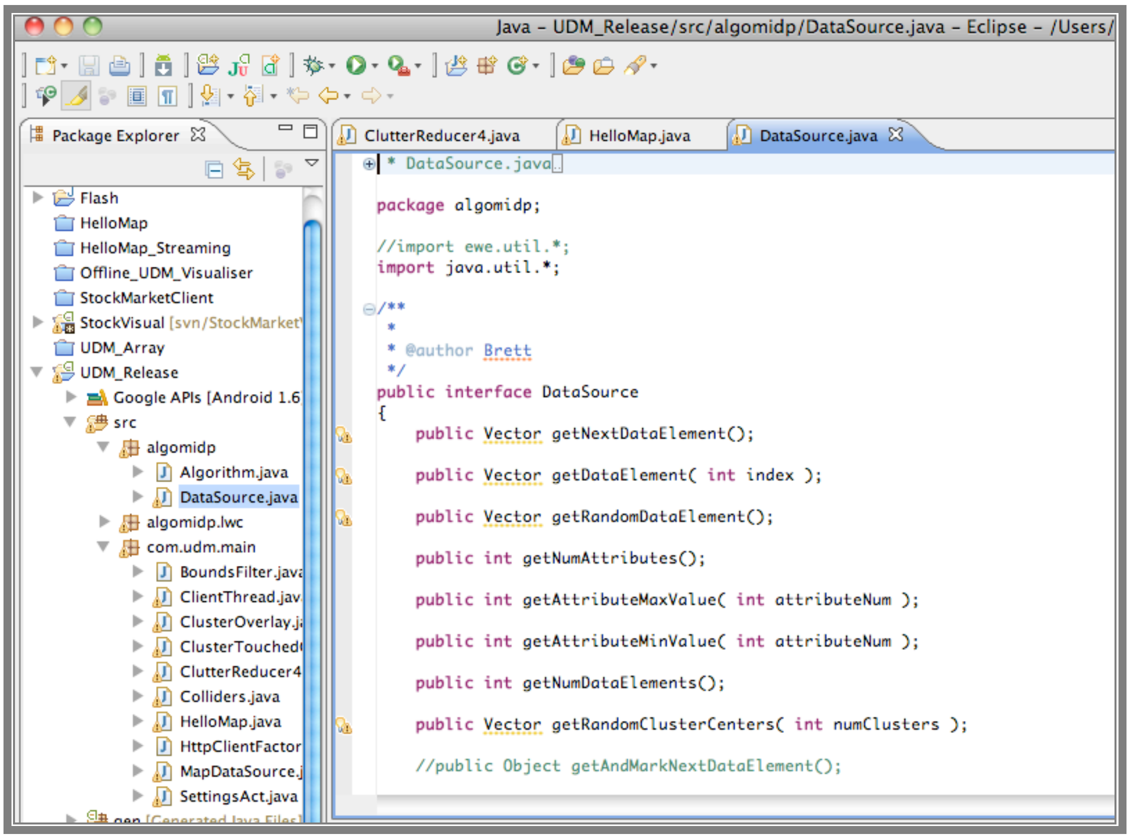


Figure 5.4 data flow in iCACV system

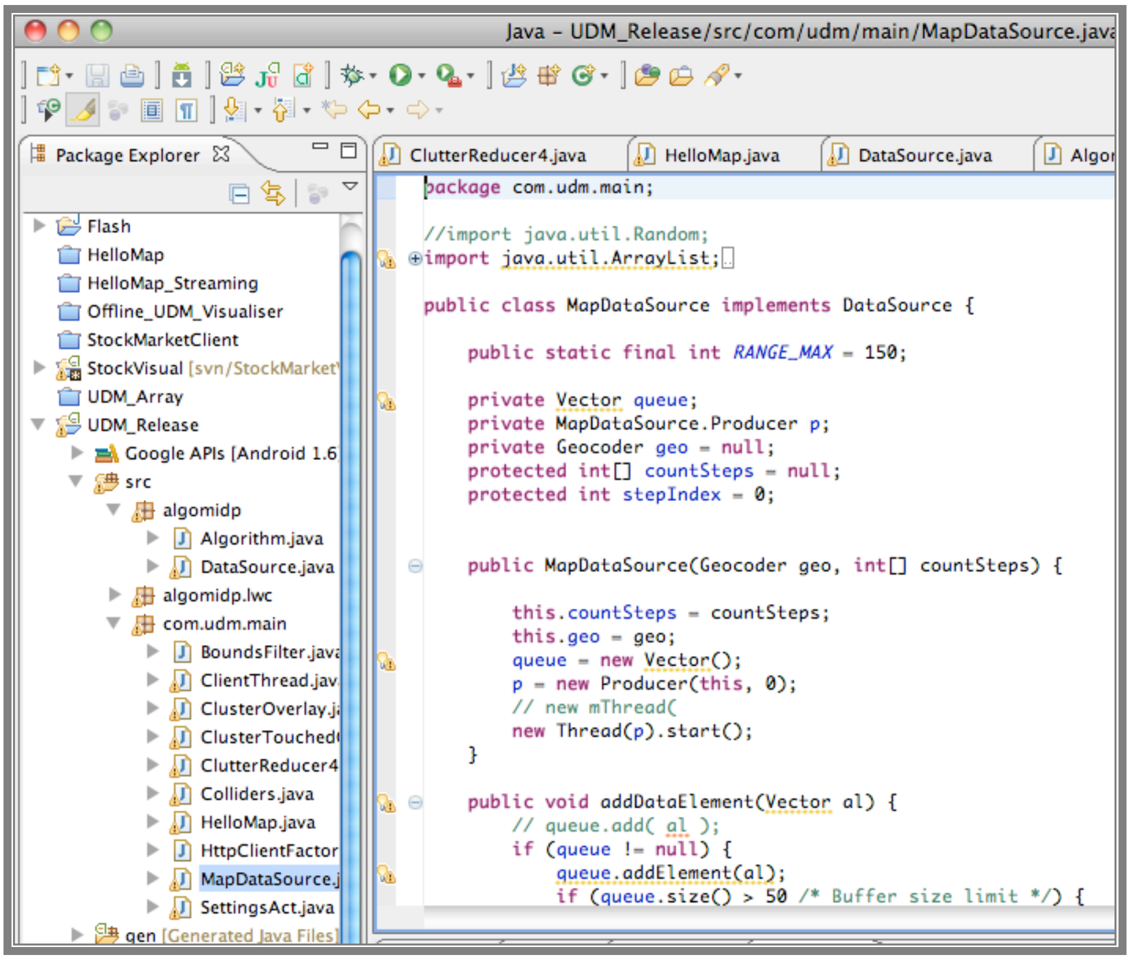
In order to make iCACV algorithm as generic as possible, the data acquisition system is left out of the main algorithm. It only provides the data streams to the algorithm. This means that the implementation of iCACV framework is independent of the data source, which enables iCACV to be used for different types of datasets.

In our implementation, we have developed a Java Interface that is named DataSource, which has all the necessary functions and attributes. In order to feed the iCACV algorithm with the appropriate data structure, this interface needs to be implemented. This makes it very easy to reuse the algorithm as long as all methods in the DataSource interface are implemented. In our case, we have implemented this interface in MapDataSource class as shown in figures 5.5 and 5.6.



```
* DataSource.java

package algomidp;

//import ewe.util.*;
import java.util.*;

/**
 *
 * @author Brett
 */
public interface DataSource
{
    public Vector getNextDataElement();

    public Vector getDataElement( int index );

    public Vector getRandomDataElement();

    public int getNumAttributes();

    public int getAttributeMaxValue( int attributeNum );

    public int getAttributeMinValue( int attributeNum );

    public int getNumDataElements();

    public Vector getRandomClusterCenters( int numClusters );

    //public Object getAndMarkNextDataElement();
```

Figure 5.5 DataSource interface, which makes iCACV generic for any dataset

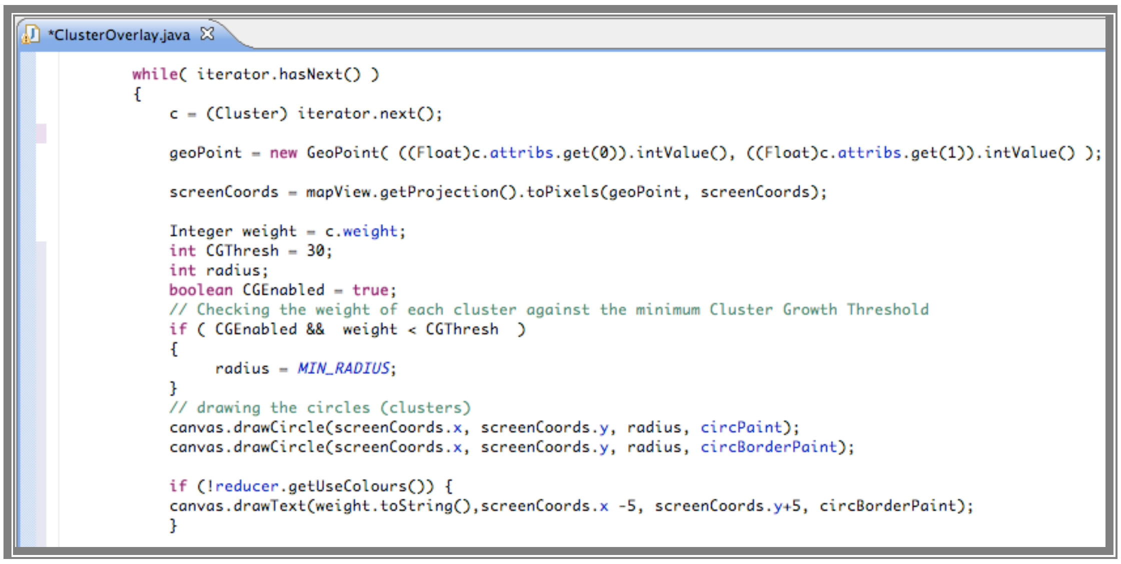


```
package com.udm.main;

//import java.util.Random;
import java.util.ArrayList;

public class MapDataSource implements DataSource {

    public static final int RANGE_MAX = 150;

    private Vector queue;
    private MapDataSource.Producer p;
    private Geocoder geo = null;
    protected int[] countSteps = null;
    protected int stepIndex = 0;


    public MapDataSource(Geocoder geo, int[] countSteps) {

        this.countSteps = countSteps;
        this.geo = geo;
        queue = new Vector();
        p = new Producer(this, 0);
        // new mThread(
        new Thread(p).start();
    }

    public void addDataElement(Vector al) {
        // queue.add( al );
        if (queue != null) {
            queue.addElement(al);
            if (queue.size() > 50 /* Buffer size limit */) {
```

Figure 5.6 MapDataSource implementation for feeding iCACV with the geo-locations (addresses) dataset

The resource awareness of iCACV is a core function that is inherited from CACV, which enables its effective operation on the resource-constrained environments of mobile phones. This can be seen as the first part of the flow chart (figure 5.4), which receives the data streams and performs the data analysis on the mobile phones. The outcome of the UDM analysis will then be sent to different modules before it is visualised to the user.

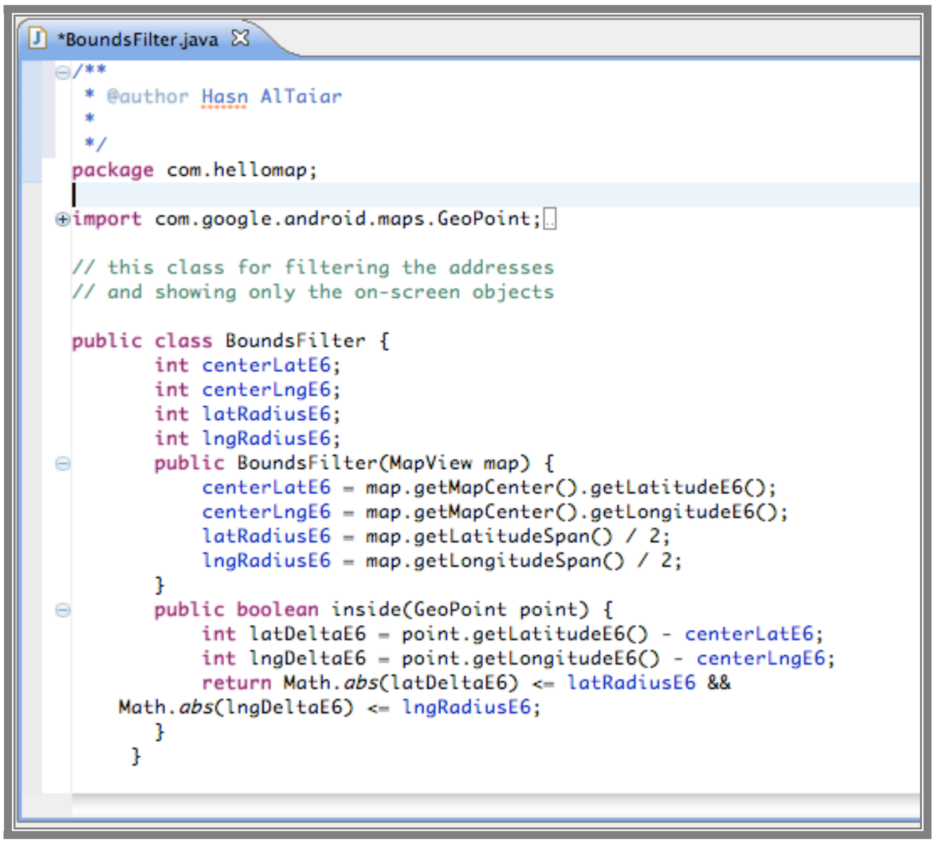
Firstly, the outcome of the UDM analysis is checked using the Cluster Growth Controller. If the cluster growth control feature were enabled, clusters would not be allowed to grow until they reach a certain threshold. This can be seen in figure 5.7, where the weight, which refers to the number of data items in each cluster, is checked before visualising it.



```
while( iterator.hasNext() )
{
    c = (Cluster) iterator.next();

    geoPoint = new GeoPoint( ((Float)c.attribs.get(0)).intValue(), ((Float)c.attribs.get(1)).intValue() );

    screenCoords = mapView.getProjection().toPixels(geoPoint, screenCoords);

    Integer weight = c.weight;
    int CGThresh = 30;
    int radius;
    boolean CGEnabled = true;
    // Checking the weight of each cluster against the minimum Cluster Growth Threshold
    if ( CGEnabled &&  weight < CGThresh  )
    {
         radius = MIN_RADIUS;
    }
    // drawing the circles (clusters)
    canvas.drawCircle(screenCoords.x, screenCoords.y, radius, circPaint);
    canvas.drawCircle(screenCoords.x, screenCoords.y, radius, circBorderPaint);

    if (!reducer.getUseColours()) {
    canvas.drawText(weight.toString(),screenCoords.x -5, screenCoords.y+5, circBorderPaint);
    }
```

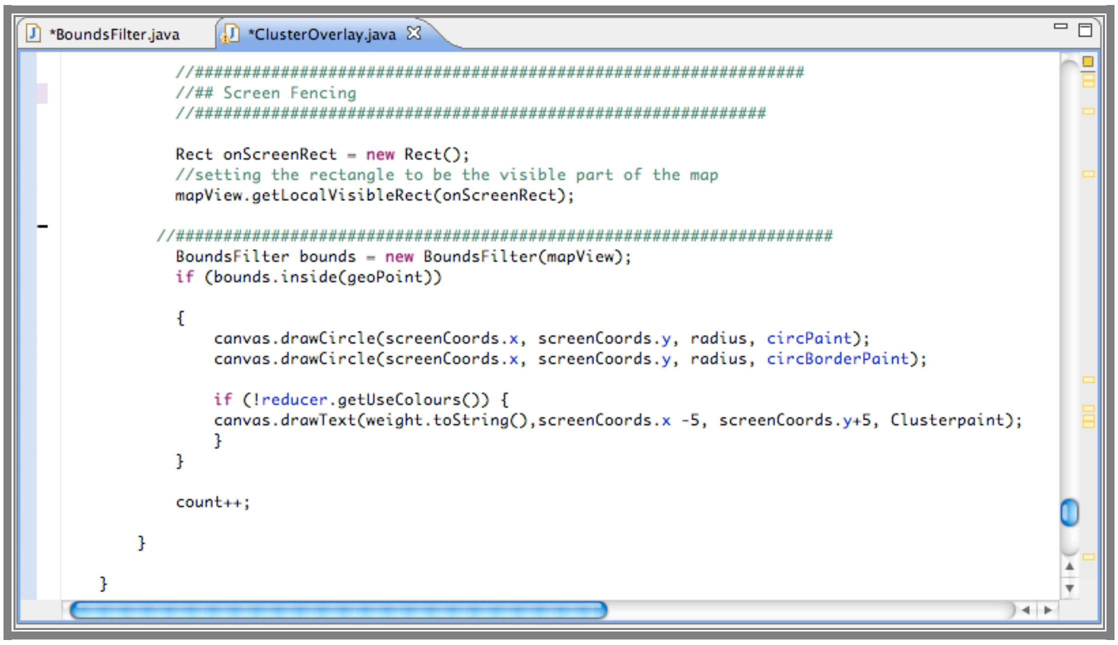
Figure 5.7 implementation of the Cluster Growth Controller

In order to implement the Screen Fencing handler, we need to have a virtual jail (virtual rectangle) on screen that represents the visible boundaries of the map. This is to compare the geographical coordinates of each single data item before the visualisation. This aims to reduce the overhead and energy that can be spent on visualising off-screen objects, which is not necessary since the user will not notice it. This virtual jail is sometimes referred to as "bucket" which is used for testing particular geo-locations. The implementation of this bucket (virtual fence) is shown in figure 5.8 and the handling of the screen fencing is shown in figure 4.9.



```
/**
 * @author Hasn AlTaiar
 *
 */
package com.hellomap;

import com.google.android.maps.GeoPoint;

// this class for filtering the addresses
// and showing only the on-screen objects

public class BoundsFilter {
       int centerLatE6;
       int centerLngE6;
       int latRadiusE6;
       int lngRadiusE6;
       public BoundsFilter(MapView map) {
           centerLatE6 = map.getMapCenter().getLatitudeE6();
           centerLngE6 = map.getMapCenter().getLongitudeE6();
           latRadiusE6 = map.getLatitudeSpan() / 2;
           lngRadiusE6 = map.getLongitudeSpan() / 2;
       }
       public boolean inside(GeoPoint point) {
           int latDeltaE6 = point.getLatitudeE6() - centerLatE6;
           int lngDeltaE6 = point.getLongitudeE6() - centerLngE6;
           return Math.abs(latDeltaE6) <= latRadiusE6 &&
    Math.abs(lngDeltaE6) <= lngRadiusE6;
       }
     }
```

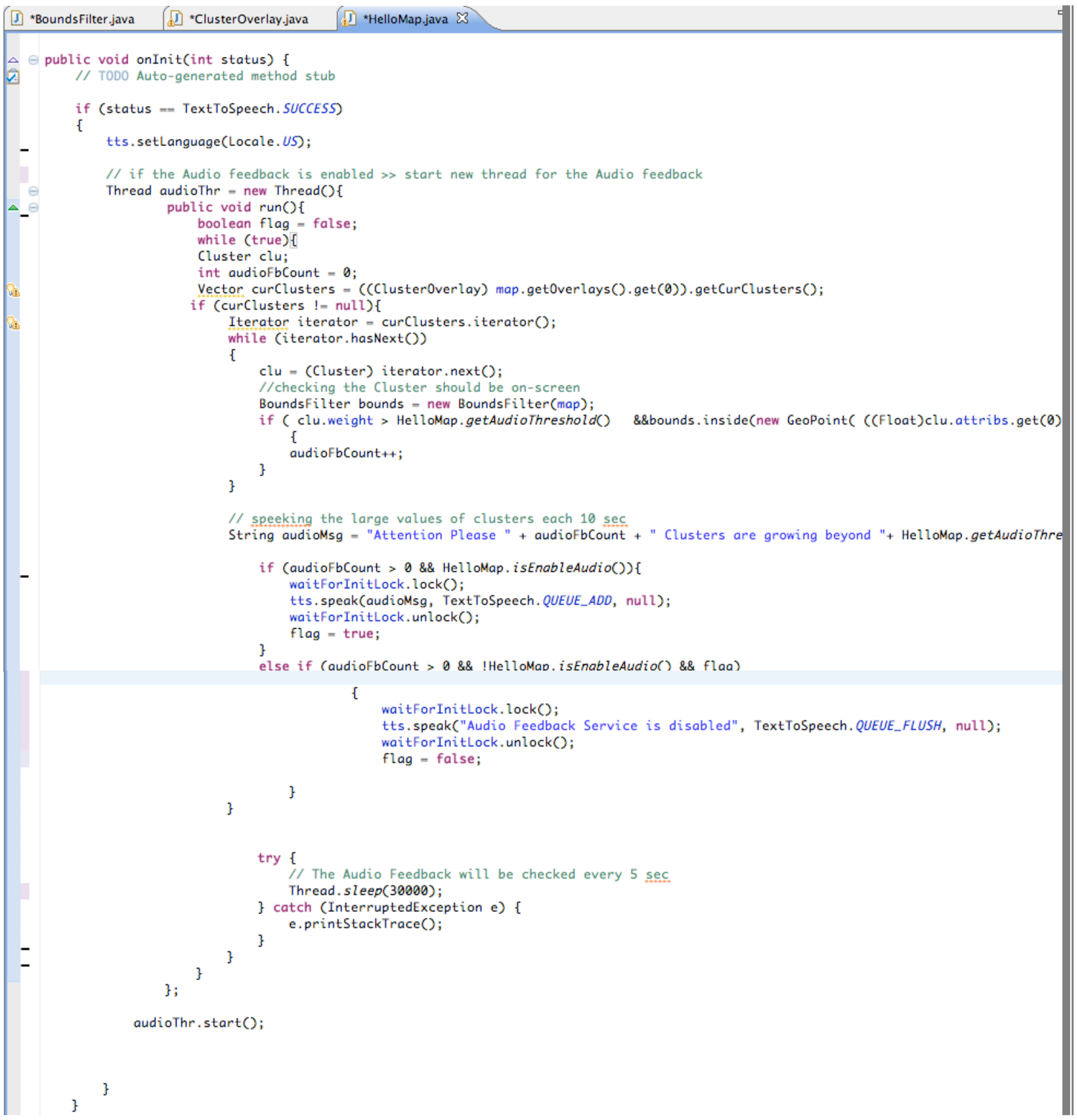
Figure 5.8 implementation of the virtual jail (bucket) that is used for screen fencing of clusters



```
            //############################################################
            //## Screen Fencing
            //#########################################################

            Rect onScreenRect = new Rect();
            //setting the rectangle to be the visible part of the map
            mapView.getLocalVisibleRect(onScreenRect);

          //#################################################################
            BoundsFilter bounds = new BoundsFilter(mapView);
            if (bounds.inside(geoPoint))

            {
                canvas.drawCircle(screenCoords.x, screenCoords.y, radius, circPaint);
                canvas.drawCircle(screenCoords.x, screenCoords.y, radius, circBorderPaint);

                if (!reducer.getUseColours()) {
                canvas.drawText(weight.toString(),screenCoords.x -5, screenCoords.y+5, Clusterpaint);
                }
            }

            count++;

        }

    }
```

Figure 5.9 the use of the bucket (virtual jail) for fencing on-screen clusters to reduce the unnecessary overhead of visualising off-screen objects

Most of the new smart phones are powered by audio libraries that support converting text into speech, and so do Google Android phones. Thus, we use the text-to-speech library that is available in Google's phone to provide an audio alert when an off-screen object reaches into a user-defined threshold (Audio Feedback threshold). This task needs to be performed in parallel with the UDM analysis and visualisation. Hence, it requires creating a dedicated thread for this reason. This thread then can be used for monitoring all clusters and then informing the user using the text-to-speech library if any of the clusters exceeds that user-defined threshold. This can be seen in figure 5.10

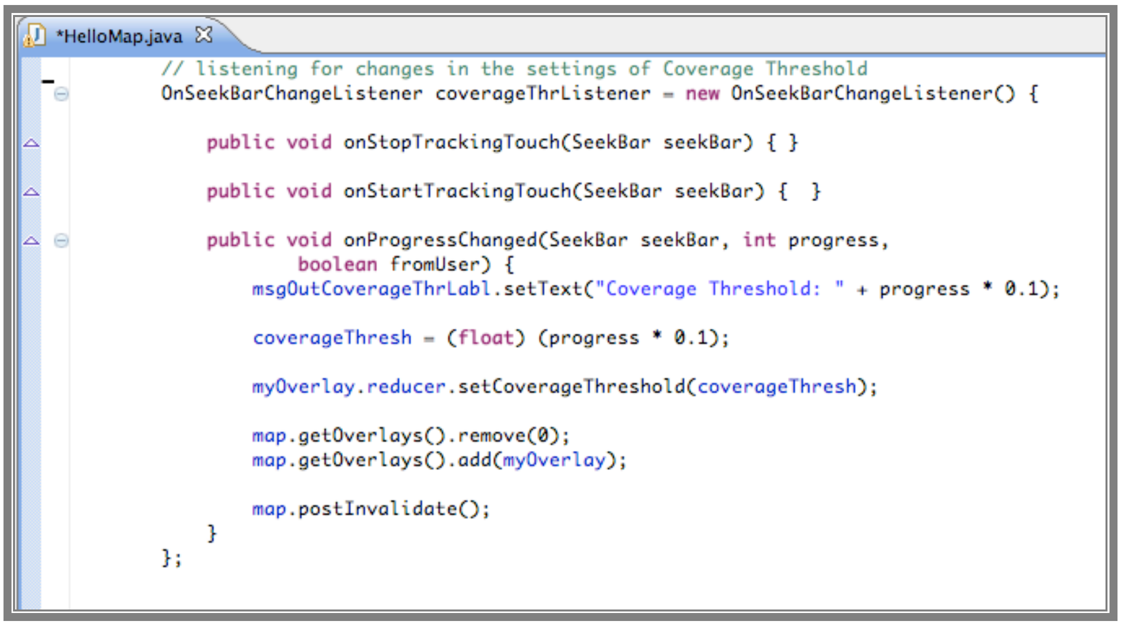


```
public void onInit(int status) {
    // TODO Auto-generated method stub

    if (status == TextToSpeech.SUCCESS)
    {
        tts.setLanguage(Locale.US);

        // if the Audio feedback is enabled >> start new thread for the Audio feedback
        Thread audioThr = new Thread(){
                public void run(){
                    boolean flag = false;
                    while (true){
                    Cluster clu;
                    int audioFbCount = 0;
                    Vector curClusters = ((ClusterOverlay) map.getOverlays().get(0)).getCurClusters();
                   if (curClusters != null){
                        Iterator iterator = curClusters.iterator();
                        while (iterator.hasNext())
                        {
                            clu = (Cluster) iterator.next();
                            //checking the Cluster should be on-screen
                            BoundsFilter bounds = new BoundsFilter(map);
                            if ( clu.weight > HelloMap.getAudioThreshold()   &&bounds.inside(new GeoPoint( ((Float)clu.attribs.get(0)
                                {
                                audioFbCount++;
                            }
                        }

                        // speeking the large values of clusters each 10 sec
                        String audioMsg = "Attention Please " + audioFbCount + " Clusters are growing beyond "+ HelloMap.getAudioThre

                            if (audioFbCount > 0 && HelloMap.isEnableAudio()){
                                waitForInitLock.lock();
                                tts.speak(audioMsg, TextToSpeech.QUEUE_ADD, null);
                                waitForInitLock.unlock();
                                flag = true;
                            }
                            else if (audioFbCount > 0 && !HelloMap.isEnableAudio() && flag)
                                        {
                                            waitForInitLock.lock();
                                            tts.speak("Audio Feedback Service is disabled", TextToSpeech.QUEUE_FLUSH, null);
                                            waitForInitLock.unlock();
                                            flag = false;

                                }
                        }


                            try {
                                // The Audio Feedback will be checked every 5 sec
                                Thread.sleep(30000);
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                        }
                    }
                };

            audioThr.start();


        }
    }
```

Figure 5.10 implementation of the Audio Feedback Controller

Moreover, it can be seen from figure 5.10 that this thread, which is dedicate for monitoring clusters and providing audio feedback, is started in oninit( ) method. This method is called once the main frame (Activity) of the application is started. As a result, the thread will start monitoring clusters on the application's initialisation. The thread checks the status of clusters each 30 seconds to assess clusters and generate a new audio alert if required. The 30 seconds time period here is necessary in two ways. First, it is beneficial for reducing the computational overhead of checking the clusters status. Second, it gives enough time for the text-to-speech service to finish the previous message to start playing any new audio alert.

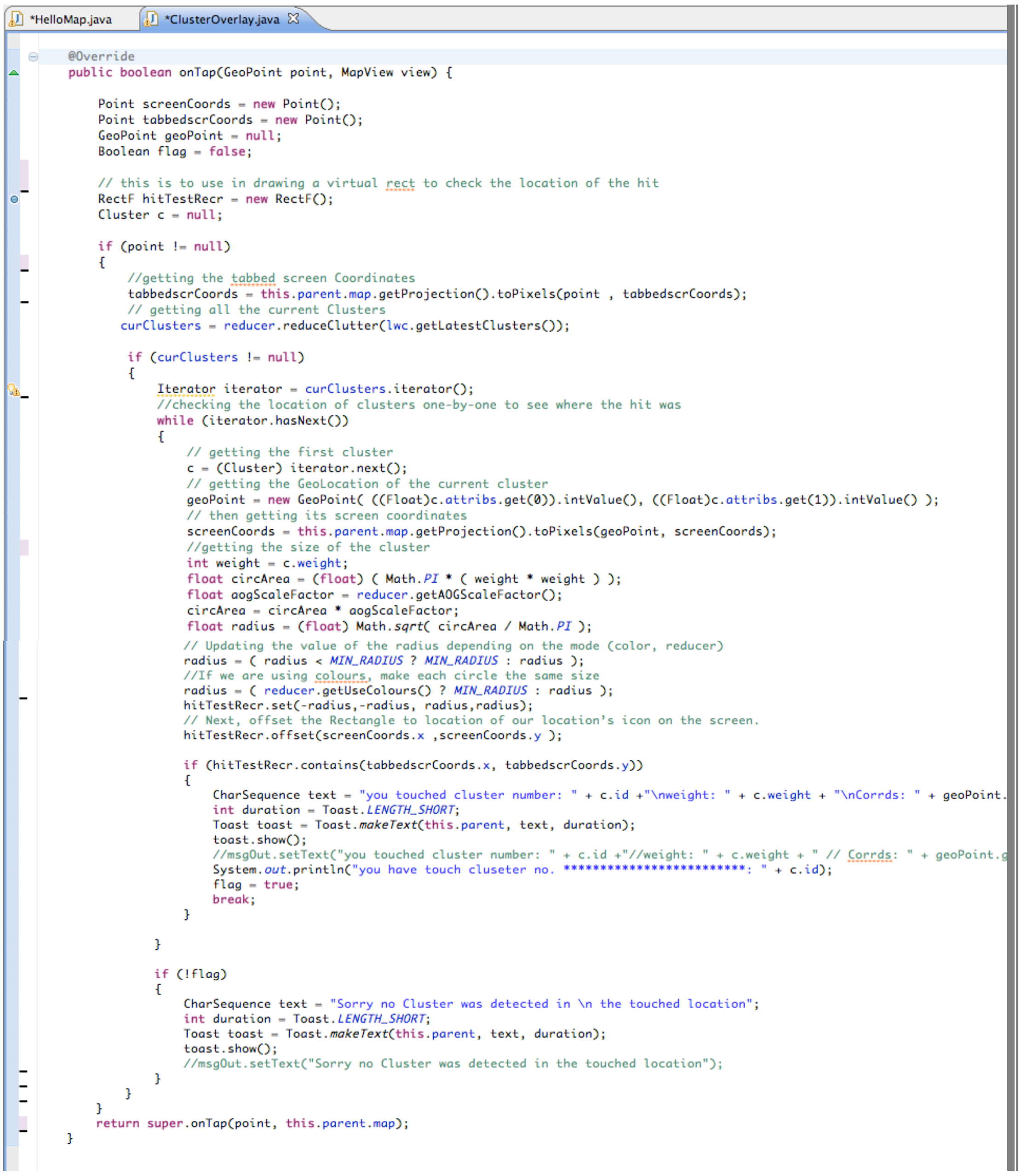
After visualising the UDM outcome on the mobile device screen, the User Preference Monitor is in charge of monitoring the user's preferences in case the user wanted to change any of the visualisation parameters or requested more information about a particular cluster. This can be implemented using different listeners that can listen to any possible input from the user and interpret that input into an appropriate value. The UDM analysis and visualisation can then dynamically be updated according to this new value/change. Figure 5.11 shows one of those listeners. This listener enables the user to change the Coverage Threshold (CT) and updates the iCACV algorithm to use this new value.



```
// listening for changes in the settings of Coverage Threshold
OnSeekBarChangeListener coverageThrListener = new OnSeekBarChangeListener() {

    public void onStopTrackingTouch(SeekBar seekBar) { }

    public void onStartTrackingTouch(SeekBar seekBar) {  }

    public void onProgressChanged(SeekBar seekBar, int progress,
            boolean fromUser) {
        msgOutCoverageThrLabl.setText("Coverage Threshold: " + progress * 0.1);

        coverageThresh = (float) (progress * 0.1);

        myOverlay.reducer.setCoverageThreshold(coverageThresh);

        map.getOverlays().remove(0);
        map.getOverlays().add(myOverlay);

        map.postInvalidate();
    }
};
```

Figure 5.11 shows the listener that controls the coverage threshold based on the user input

Furthermore, the User Preference Monitor also listens to any possible request from the user demanding more information about a particular cluster. In such cases, the User Preference Monitor will respond by examining the cluster that attracted the user's attention, and then provide an output message to inform the user appropriately about that cluster. We refer to this as Selective Focusing and sometimes it is also called "details-on-demand". In implementing this, we create a bucket (virtual rectangle) as explained earlier to check the geo-location (coordinates) of the touched/tabbed/clicked location in order to identify the interesting cluster. By interesting here, we mean the one that the user wants to know more information about. This can be seen in figure 5.12, where we examine any possible input from the user to identify the interesting cluster to provide detail information.


*HelloMap.java   *ClusterOverlay.java

```java
    @Override
    public boolean onTap(GeoPoint point, MapView view) {

        Point screenCoords = new Point();
        Point tabbedscrCoords = new Point();
        GeoPoint geoPoint = null;
        Boolean flag = false;

        // this is to use in drawing a virtual rect to check the location of the hit
        RectF hitTestRecr = new RectF();
        Cluster c = null;

        if (point != null)
        {
            //getting the tabbed screen Coordinates
            tabbedscrCoords = this.parent.map.getProjection().toPixels(point , tabbedscrCoords);
            // getting all the current Clusters
           curClusters = reducer.reduceClutter(lwc.getLatestClusters());

            if (curClusters != null)
            {
                Iterator iterator = curClusters.iterator();
                //checking the location of clusters one-by-one to see where the hit was
                while (iterator.hasNext())
                {
                    // getting the first cluster
                    c = (Cluster) iterator.next();
                    // getting the GeoLocation of the current cluster
                    geoPoint = new GeoPoint( ((Float)c.attribs.get(0)).intValue(), ((Float)c.attribs.get(1)).intValue() );
                    // then getting its screen coordinates
                    screenCoords = this.parent.map.getProjection().toPixels(geoPoint, screenCoords);
                    //getting the size of the cluster
                    int weight = c.weight;
                    float circArea = (float) ( Math.PI * ( weight * weight ) );
                    float aogScaleFactor = reducer.getAOGScaleFactor();
                    circArea = circArea * aogScaleFactor;
                    float radius = (float) Math.sqrt( circArea / Math.PI );
                   // Updating the value of the radius depending on the mode (color, reducer)
                   radius = ( radius < MIN_RADIUS ? MIN_RADIUS : radius );
                   //If we are using colours, make each circle the same size
                   radius = ( reducer.getUseColours() ? MIN_RADIUS : radius );
                   hitTestRecr.set(-radius,-radius, radius,radius);
                   // Next, offset the Rectangle to location of our location's icon on the screen.
                   hitTestRecr.offset(screenCoords.x ,screenCoords.y );

                   if (hitTestRecr.contains(tabbedscrCoords.x, tabbedscrCoords.y))
                   {
                       CharSequence text = "you touched cluster number: " + c.id +"\nweight: " + c.weight + "\nCorrds: " + geoPoint.
                       int duration = Toast.LENGTH_SHORT;
                       Toast toast = Toast.makeText(this.parent, text, duration);
                       toast.show();
                       //msgOut.setText("you touched cluster number: " + c.id +"//weight: " + c.weight + " // Corrds: " + geoPoint.g
                       System.out.println("you have touch cluseter no. ************************: " + c.id);
                       flag = true;
                       break;
                   }

               }

               if (!flag)
               {
                   CharSequence text = "Sorry no Cluster was detected in \n the touched location";
                   int duration = Toast.LENGTH_SHORT;
                   Toast toast = Toast.makeText(this.parent, text, duration);
                   toast.show();
                   //msgOut.setText("Sorry no Cluster was detected in the touched location");
               }
           }
        }
        return super.onTap(point, this.parent.map);
    }
```

Figure 5.12 implementation of the User Preference Manager and how it responds to any user input

After the implementation of iCACV model and developing the framework, a statistical and systematic evaluation is required to examine the effectiveness of incorporating all these techniques to the original CACV model. Thus, the evaluation will be our main topic for the next section.

5.3 Datasets and Evaluation Strategies

In order to evaluate the proposed model (iCACV), we need to design some scenarios and testing cases to apply them for both models (CACV and iCACV). By comparing the results of both models, we can measure the improvements of iCACV. Therefore, such task demands a full functioning and operational environment, where the models (iCACV and CACV) are being fed with datasets and practical results can then be extracted. Moreover, evaluating data visualisation can be very difficult since people are uniquely different in their ways of looking at things and comprehending ideas. For instance, one graph can be very clear for some people, while it could make no sense for some others. For such reasons, statistical and systematic measurements are highly preferred in such cases to ensure appropriate assessment of any potential improvements. Hence, following this, we introduce a case study to test both models. Before presenting the case study, we provide a brief overview of our experimental system. A list of the components that were used for evaluation will then be presented along with the datasets.

5.3.1 Experimental setup

For evaluating both models (CACV and iCACV), we have designed an architecture that simulates a real scenario of feeding data items at a very high speed to be analysed and visualised on mobile devices. This section presents the architecture, main components and the datasets that were used for evaluating the prototypes.

Basically, our system architecture consists of components that are necessary for such applications, which include a smart phone, Apache server, Data Streamer framework, MySql database, and other components. Details of these components are listed in the next section. Figure 5.13 shows the main components of our experimental system architecture.

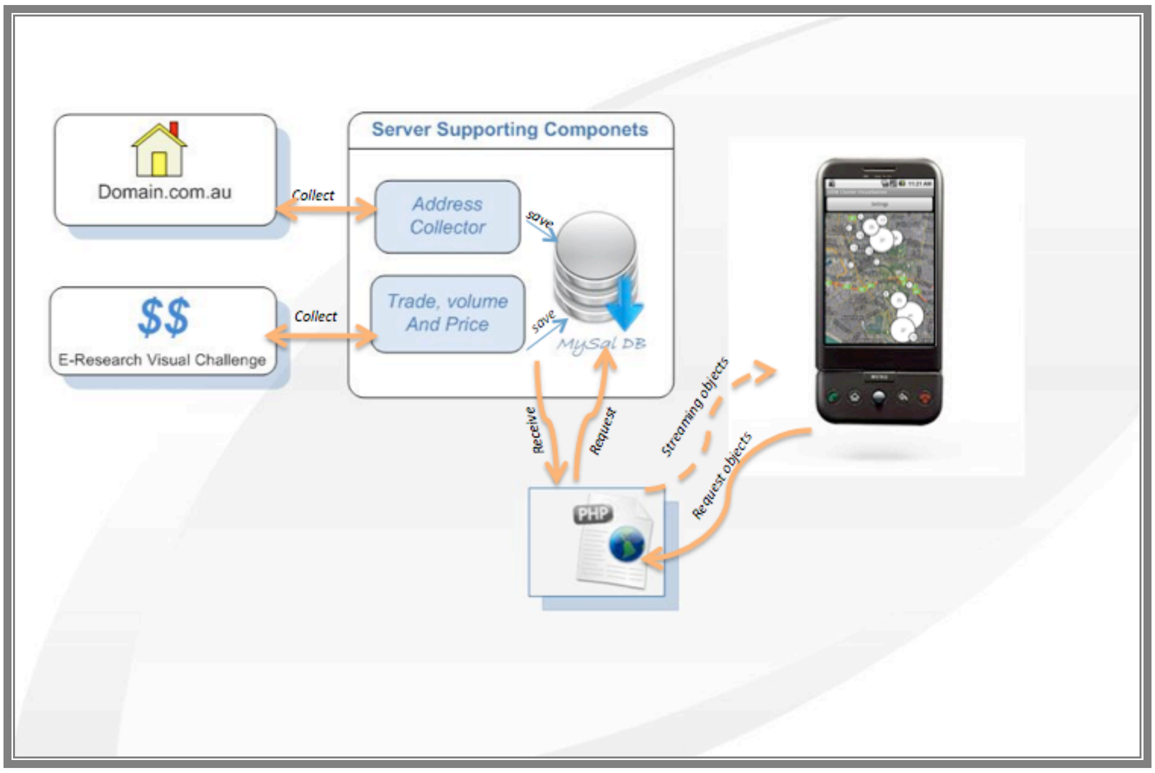


Figure 5.13 the main components of the experimental system architecture

It can be seen from figure 5.13 that our architecture has two data sources, which are domain.com.au and the E-Research Visualisation Challenge. Crawler applications were implemented to visit the data sources and collect the evaluation data. The crawler applications were implemented in java using java.net and java.net.URIConnection packages. The main use of these crawler applications is to visit the data sources and collect data to store it in a reliable and accessible location (MySql database). Eventually, this data is used for evaluating the CACV and iCACV models by streaming a continuous flow of data to a mobile phone, where those streams are analysed and visualised as shown in figure 5.13.

Following this, we conduct five main experiments to examine the effectiveness of each interaction feature of the iCACV. First, we explain what is measured and the mechanism for doing so at the beginning of each experiment. For most of the experiments the real estate dataset (addresses) was used to generate the data streams as the UDM visualisation input. The stock market dataset was only used for demonstrating the Selective Focusing function and the Audio Feedback capabilities of iCACV. Using these two datasets, which are totally different, demonstrates the fact that our proposed iCACV model is generic and can be used for analysing and visualising any type of dataset.

The main scenario of our experiments is an emergency call monitoring system, where a mobile phone is set up to receive the locations of moving vehicles and those received locations should be analysed and visualised on that portable device. Those received geo-locations should be clustered and visualised on a map on the mobile screen in order to help the user in learning about what is happening in real-time. The clusters that are drawn on screen may grow or shrink depending on the number of data items that are placed in each cluster in real-time. Such application can be very useful in many different scenarios where this visualisation of the UDM outcomes is essentially required. Examples of such scenarios include:

- An emergency call centre has received a report that there is a catastrophic fire that has just started on in a particular suburb. A group of fire fighters are deployed into the field to control the situation. The call centre started to receive more calls reporting that this fire is spreading into different suburbs. The fire fighters leaders need to know in real-time which areas have been affected most to deploy more aiders and equipments to those places. Instead of calling the call centre, which can be in a chaos in such situations, our iCACV can do the analysis and visualise the outcome of clustering the phone calls based on the geo-location (address) of the callers and display it on the mobile phones of the fire fighters leaders to help them make well-informed decisions in real-time.
- Another scenario where our iCACV model can serve users in a similar way is that public transport officers are deployed into the streets. For some reasons (disasters, or other reasons), few train lines are not operating on that day and the transport officers are receiving lots of phone calls demanding taxi services. In such cases, the Public Transport officers need to know in real-time where is the highest demand on the taxi services, in which suburb? And what is the demand, exactly? What is the rate of change? And all these types of questions, to make appropriate decisions of sending a certain number of cabs into this suburb or that. This would be based on the visualisation of the UDM analysis that can be performed on the officer’s mobile phone.

5.3.2 Evaluation Datasets

To test our iCACV model, a high-speed data stream is needed to be fed to the iCACV model. For that, we have collected data from different sources to show the functionality of the proposed model (iCACV). The first dataset consists of real estate addresses. These addresses were collected from a real estate website, namely (domain.com.au). We have developed an application that visits this website frequently and collect data for different suburbs. The main use of this data is to show the clustering and visualisation functionalities of the iCACV system.

The other source of data was also a real data set, obtained from the E-Research Visualisation Challenge[10] and is over 1 GB of trade, volume and price information for stock market data that we stream to the mobile device as if it were a live stock feed from a remote server. As we mentioned, this was also a real dataset, which was collected from the stock market for around 1 year period of time and made available for public as part of the 2009 visualisation Challenge by E-Research Australia. By streaming the stock market dataset, we analyse it and visualise the outcome on the mobile phone based on the price and volume values to give users a good indication of what is happening in real-time

For both datasets, we have developed supporting components that synthesise real-life scenarios by streaming the data records into both the CACV and iCACV models. As a result, we have a continuous flow of data streams that can be analysed on any portable device in an attempt to simulate a real scenario of a real-time traffic monitoring system.

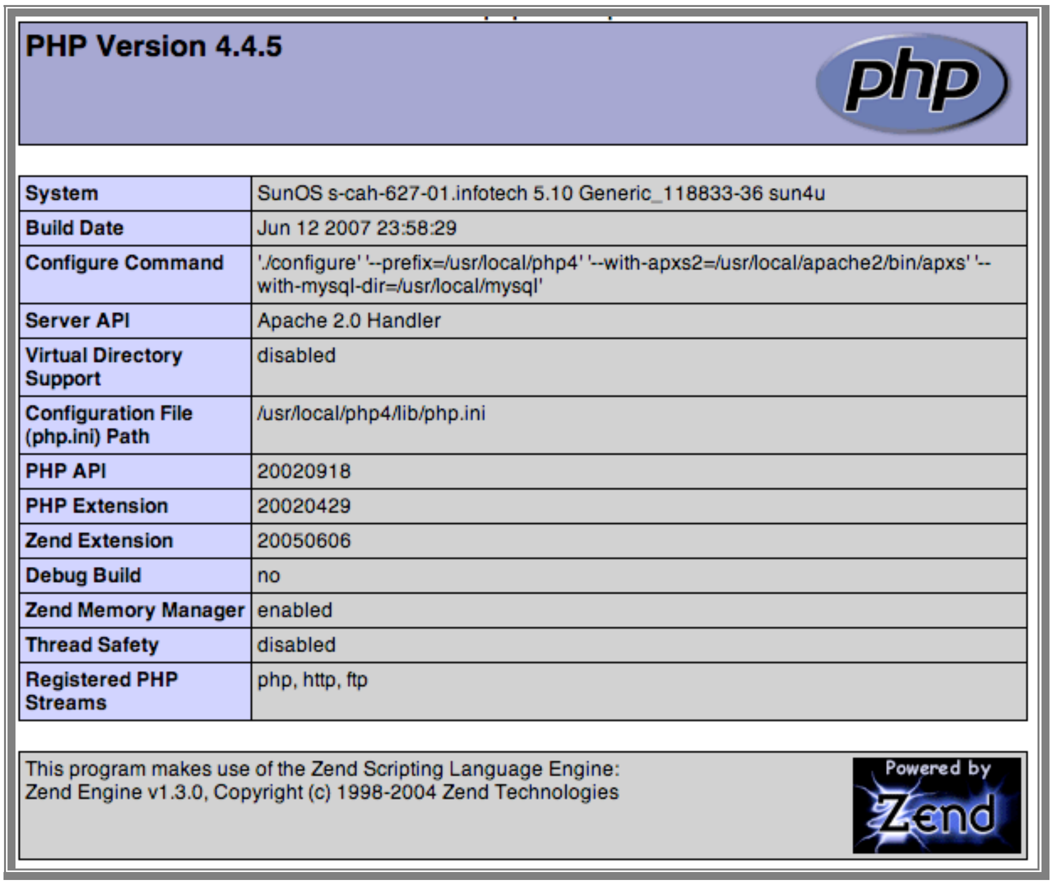
5.3.3 Main Components

- **Apache Server**

In order to feed the mobile device with a continuous stream of data items, the datasets need to be uploaded into an accessible web server, where they can be accessed using some sockets or http channels. For this reason, we have placed the datasets on an Apache[11] server and developed a PHP script that feeds those datasets as streams of data items to any requesting device. The Apache Server information is shown in figure 5.14.

[10] E-Research Australia: Visualisation Challenge http://www.eresearch.edu.au/vischallenge
[11] Details of Apache server can be found at http://www.apache.org/

Apache is also an open source application that offers a nice functionality for free with a stable support for PHP programs.



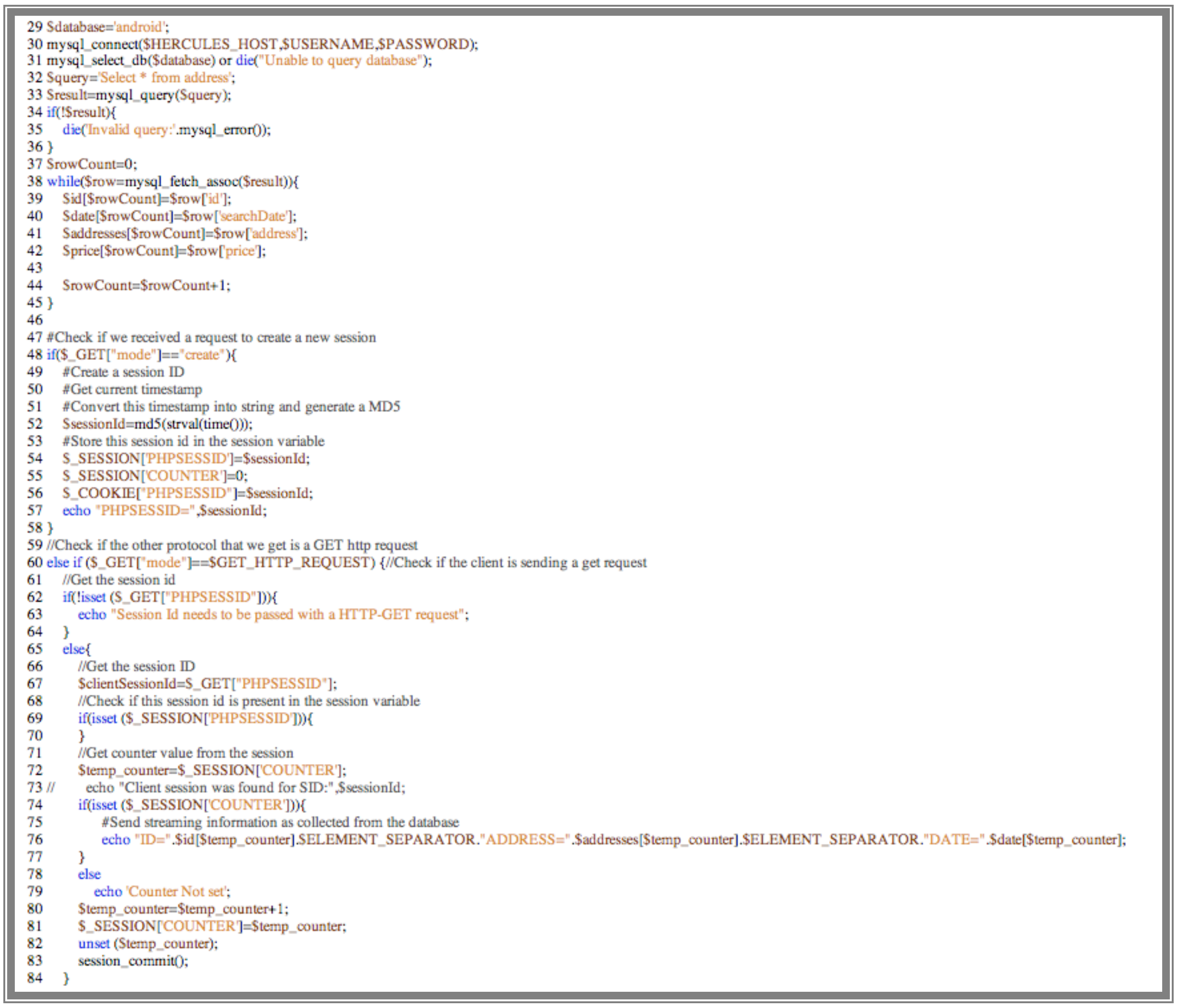
PHP Version 4.4.5

System	SunOS s-cah-627-01.infotech 5.10 Generic_118833-36 sun4u
Build Date	Jun 12 2007 23:58:29
Configure Command	'./configure' '--prefix=/usr/local/php4' '--with-apxs2=/usr/local/apache2/bin/apxs' '--with-mysql-dir=/usr/local/mysql'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/php4/lib/php.ini
PHP API	20020918
PHP Extension	20020429
Zend Extension	20050606
Debug Build	no
Zend Memory Manager	enabled
Thread Safety	disabled
Registered PHP Streams	php, http, ftp

This program makes use of the Zend Scripting Language Engine:
Zend Engine v1.3.0, Copyright (c) 1998-2004 Zend Technologies

Figure 5.14 information of the Apache Server that is used for hosting the evaluation datasets

Moreover, a PHP script was also required to enable mobile phones to access the datasets in a form of data streams in order to simulate a real scenario of streaming addresses from a call centre to be visualised on mobile phones. Figure 5.15 demonstrates the main parts of the PHP script that is used for this purpose. It can be noticed that each data item consists of address, date of collecting that record, and the suburb. Also, in order to enable these records to be transferred as a stream of data, it was required to arrange for sessions between the requesting mobile phone and the server. This can be seen in the figure 5.15.



```php
$database='android';
mysql_connect($HERCULES_HOST,$USERNAME,$PASSWORD);
mysql_select_db($database) or die("Unable to query database");
$query='Select * from address';
$result=mysql_query($query);
if(!$result){
   die('Invalid query:'.mysql_error());
}
$rowCount=0;
while($row=mysql_fetch_assoc($result)){
   $id[$rowCount]=$row['id'];
   $date[$rowCount]=$row['searchDate'];
   $addresses[$rowCount]=$row['address'];
   $price[$rowCount]=$row['price'];

   $rowCount=$rowCount+1;
}

#Check if we received a request to create a new session
if($_GET["mode"]=="create"){
   #Create a session ID
   #Get current timestamp
   #Convert this timestamp into string and generate a MD5
   $sessionId=md5(strval(time()));
   #Store this session id in the session variable
   $_SESSION['PHPSESSID']=$sessionId;
   $_SESSION['COUNTER']=0;
   $_COOKIE["PHPSESSID"]=$sessionId;
   echo "PHPSESSID=",$sessionId;
}
//Check if the other protocol that we get is a GET http request
else if ($_GET["mode"]==$GET_HTTP_REQUEST) {//Check if the client is sending a get request
   //Get the session id
   if(!isset ($_GET["PHPSESSID"])){
      echo "Session Id needs to be passed with a HTTP-GET request";
   }
   else{
      //Get the session ID
      $clientSessionId=$_GET["PHPSESSID"];
      //Check if this session id is present in the session variable
      if(isset ($_SESSION['PHPSESSID'])){
      }
      //Get counter value from the session
      $temp_counter=$_SESSION['COUNTER'];
//       echo "Client session was found for SID:",$sessionId;
      if(isset ($_SESSION['COUNTER'])){
         #Send streaming information as collected from the database
         echo "ID=".$id[$temp_counter].$ELEMENT_SEPARATOR."ADDRESS=".$addresses[$temp_counter].$ELEMENT_SEPARATOR."DATE=".$date[$temp_counter];
      }
      else
         echo 'Counter Not set';
      $temp_counter=$temp_counter+1;
      $_SESSION['COUNTER']=$temp_counter;
      unset ($temp_counter);
      session_commit();
   }
```

Figure 5.15 PHP script that provides the data streams

- ### MySql Database

A database server was required to store the massive amount of data that we collected from the two data sources. A MySql[12] server is used for this purpose since it offers a good and stable functionality and it is very compatible with Apache server, which hosts our data streamer application (PHP Script). MySql is also available for general use for free under the GPL License[13].

- ### Crawler Applications

As can be seen from figure 5.13 that was shown earlier, a crawler application was required to collect data from the data sources and store it in a stable and accessible

[12] Details of MySql Server can be found at http://www.mysql.com/
[13] MySQL General Public Licensing policy http://www.mysql.com/about/legal/licensing/index.html

location such as our MySql server. We have developed a java-based application that frequently visits Domain.com.au to collect real estate addresses for random suburbs within the city of Melbourne (in Australia) and then store them into the database. We also store some other fields with the address such as the date of collecting the data in order to facilitate tracking the data in case it is needed in the future.

HTTP Client libraries were used in Java to access the data from the data sources through the use of java.net.URLConnection class. Generally, Java language offers a very stable and attractive platform for developing such applications and it is very compatible with Apache and MySql Servers.

5.3.4 Results Analysis

We have explained earlier the experimental system architecture and the main components. Also, the datasets that can be fed to the UDM visualisation models were presented in the previous sections. In order to statistically measure the implication of each module in the iCACV framework, we now need to design different experiments to assess the effectiveness of both CACV and iCACV. When attaining the results of both models for the same scenario, we can compare and contrast them to demonstrate the effectiveness of our proposed model (iCACV).

As explained earlier, we have implemented few server-side components that can stream data addresses at a very high speed to simulate a real scenario of receiving a large number of phone calls and then clustering and visualising them on mobile phones. We have done similar experiments using these datasets on both models (CACV and iCACV) to demonstrate possible improvements. In the next sub-sections, we review each main experiment, the dataset that was used, the evaluation variable, and the gain of each case.

- **Effectiveness of Dynamic Setting of UDM Visualisation parameters**

The first main experiment aims to examine the implication of providing a user-friendly control panel that enables the user to personalise the UDM visualisation based on his/her parameters settings. For conducting this experiment, we use the real estate dataset in the same scenario that we have explained earlier in this section. The same experiment was executed five times on each one of the two models (CACV and iCACV) to attain results that are as much appropriate as possible. The results of the five experiments were averaged to produce more precise results. We indicate the implication of the iCACV interaction by continuously measuring the coverage percentage of the screen. This percentage explains statistically the amount of clutter on the screen.

Figure 5.16 shows the percentage of clutter (Coverage Percentage) against the number of objects that were analysed and visualised on both CACV and iCACV. It can be seen that the algorithm switches dynamically to different visualisation mode when enabling the dynamic visualisation settings in the iCACV. This produces dynamically adaptive clutter percentage based on the user's input.

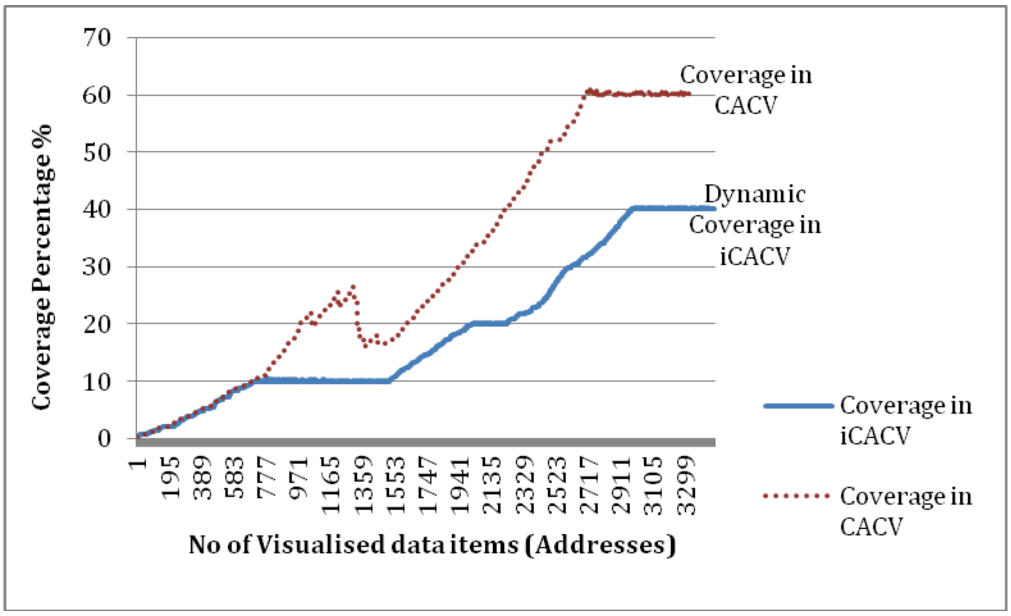


Figure 5.16 percentage of clutter vs. the number of visualised objects in CACV and iCACV

From figure 5.16, it can be seen that the CACV system did not switch to the colouring mode, which reduces the percentage of cluttered screen, until it visualised more than 2600 objects. In contrast, after our improvement, figure 5.16 shows that the algorithm (iCACV) dynamically switched to colouring mode after visualising roughly 800 objects of the same dataset. This change was correspondent to the user's action during run-time. Moreover, when the user accepted to tolerate higher percentage of cluttered screen by increasing the coverage threshold (at run-time), the algorithm also dynamically adapted itself to switch the visualisation back to the normal or scaling mode. This is shown in figure 5.16 at approximately 1600 objects, where the percentage of cluttered screen started to increase again in correspondence to the user's preference of tolerating higher clutter percentage.

In summary, using interactivity, our iCACV model has enabled the user to be in full control of the visualisation process according to the current context. By context here, we mean the current screen space, resolution, level of available resources as well as the user's preference and clutter-tolerance. This shows that our user-driven visualisation mechanism has resulted less clutter, and better visualization.

• Effectiveness of Interaction for better Cluster Distinctiveness

Like the previous experiment, this one uses the real estate dataset (addresses) that is fed as a high speed data streams to the mobile phone in a similar experimental scenario. Similarly, we have also run the experimental system for five times to assure the appropriateness of the results. This experiment aims to assess the User Preference Monitor of our framework (iCACV) and checks its responsiveness to any user input. We examine the capability of the user to differentiate clusters, change their layout, and control the clusters distinctiveness at run-time as an indicator of better of poorer visualisation.

The second part of this experiment is dedicated to evaluating the Selective Focusing interaction capability that we proposed as part of the iCACV. To do that, we run the same experiment on the iCACV using both datasets (addresses and stock market data). When running this experiment, we aim to get the framework to provide more information about any chosen cluster. The reason for conducting the same experiment using the two datasets is to demonstrate that our model (iCACV) offers a generic UDM visualisation that works efficiently regardless of the data source.

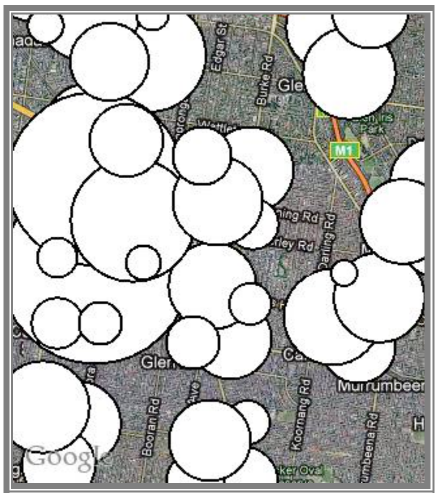
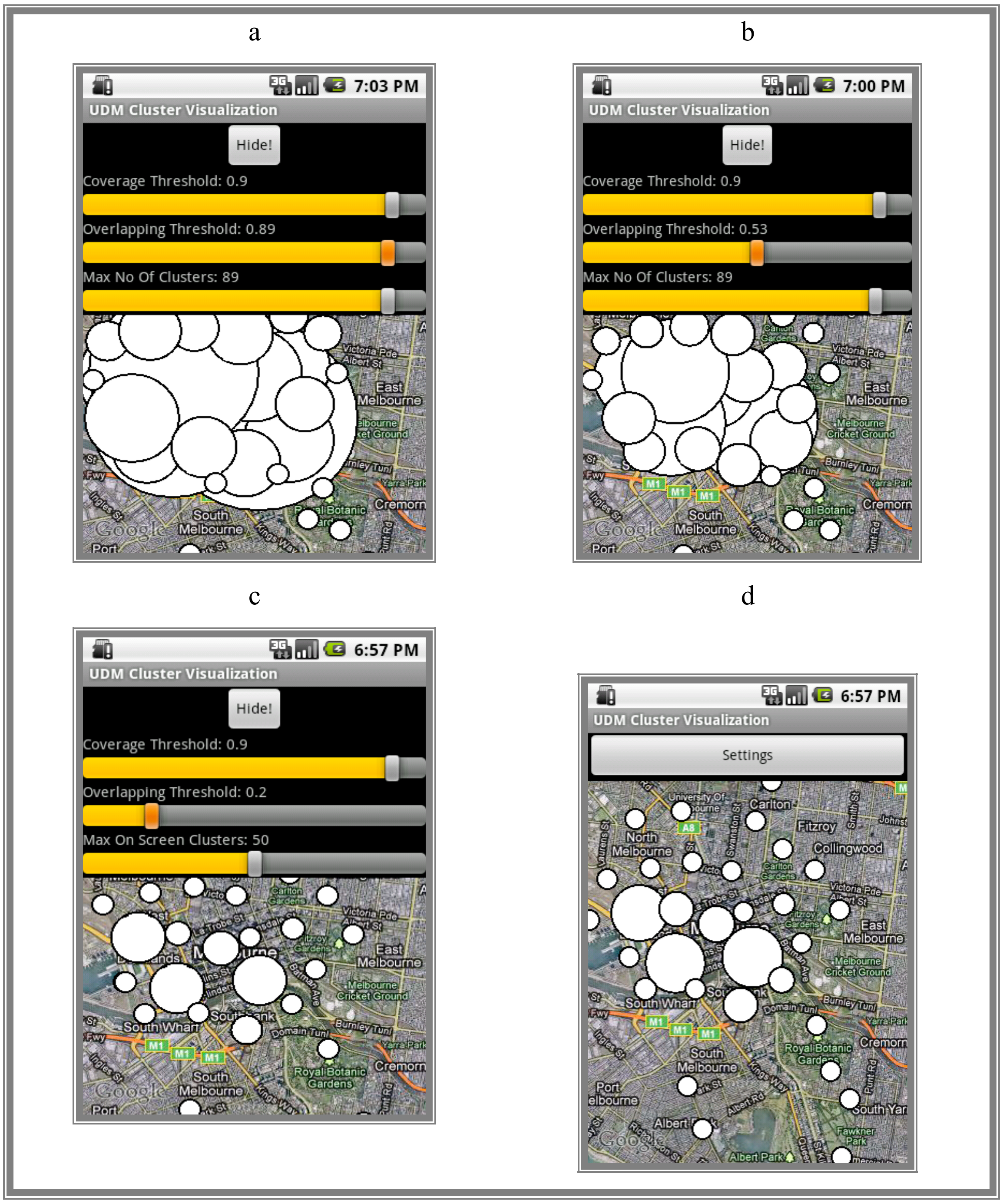


Figure 5.17 clusters in the normal visualisation mode of CACV

As we have shown earlier in chapter 3 (and in figure 5.17), the user has no control on what is displayed on screen, not on how the clusters are laid out. This can cause clarity and accuracy issues in UDM visualisation, which could defeat the main goal of implementing such application. Therefore, when incorporating interaction to the CACV algorithm, iCACV offers a personalised framework of UDM visualisation according to the user's preferences at run-time. This enables the user to be in full control of the clusters that are drawn on the screen. It also assures the user's understanding/comprehension of the presented data as in figure 5.18.

Figure 5.18 shows the true implementation of the user-driven approach of UDM visualisation, where the user is given the capability to control how the clusters are drawn and can change it at any point of time. This ensures that the user can change the data visualisation process to differentiate clusters and make more sense of the presented data.



a: 7:03 PM — Coverage Threshold: 0.9; Overlapping Threshold: 0.89; Max No Of Clusters: 89

b: 7:00 PM — Coverage Threshold: 0.9; Overlapping Threshold: 0.53; Max No Of Clusters: 89

c: 6:57 PM — Coverage Threshold: 0.9; Overlapping Threshold: 0.2; Max On Screen Clusters: 50

d: 6:57 PM

Figure 5.18 adaptation of the clustering algorithm to the dynamically changing overlapping-percentage to improve clusters distinctiveness

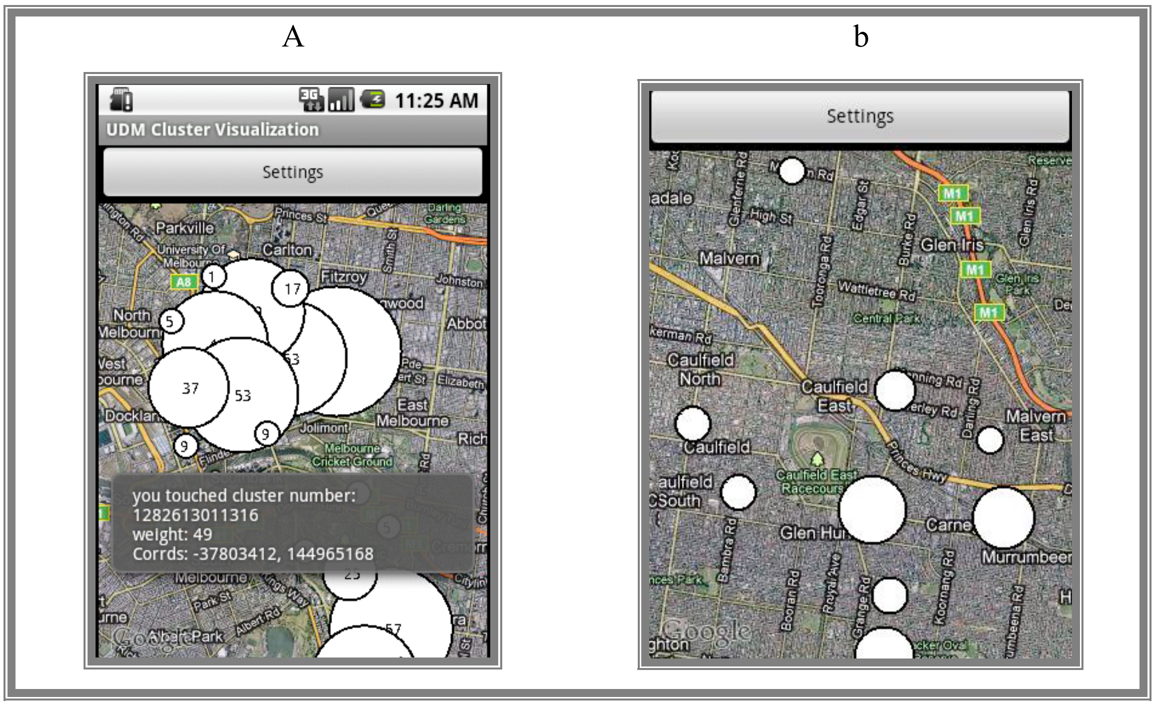
In brief, the results of the first section of this experiment are shown in figure 5.18. The graph shows that users have full control of the clusters distinctiveness level in iCACV. The screenshots demonstrate that the level of clusters' distinctiveness is changing according to the user’s input. This enables users to accurately differentiate the visualized clusters.

The second section of the experiment includes running the similar scenario and feeding both datasets (the addresses and the stock market data) to the iCACV to assess the Selective Focusing capability and to learn how the User Preference Monitor can handle users request about any cluster. As we have shown the implementation of the Selective Focusing earlier in this chapter, we demonstrate the functionality of this interaction feature and how it can support the user’s decision-making process.

Figure 5.19 (a) shows the Selective Focusing “details-on-demand” on the iCACV when feeding the real estate dataset (addresses). It assures that users are given the accurate information about any cluster, as they require. Tapping any of the clusters gives users detailed information about that touched/tapped cluster. Also, in case the audio feedback was enabled, users will also be given an audio alert telling the correspondent cluster information. We have also developed a demonstration video that can be viewed at:

http://www.youtube.com/user/callmerahul20#p/a/u/2/u7GthmR7OXc

Furthermore, we have also given the user the control to change the preference of the maximum number of clusters required on screen as shown in figure 5.19 (B). This enables the user to focus on limited number of clusters if the amount of clutter were high to clarify the data presentation.



you touched cluster number:
1282613011316
weight: 49
Corrds: -37803412, 144965168

Figure 5.19 (a) Selective Focusing interaction
(b) dynamic adaptation to the desired maximum number of clusters

On the other hand, to demonstrate how generic iCACV model is, we feed the stock market dataset to the mobile phone to be analysed and visualised onboard. Supporting components were developed to stream the dataset to the mobile phone so that to simulate a real data streaming application as if it were coming from a real-time monitoring system. The implementation of this experiment using the stock market dataset is shown in figure 5.20. This figure demonstrates the Selective Focusing interaction capability when using iCACV model to analyse and visualise the stock market dataset on the Google Android mobile phone.

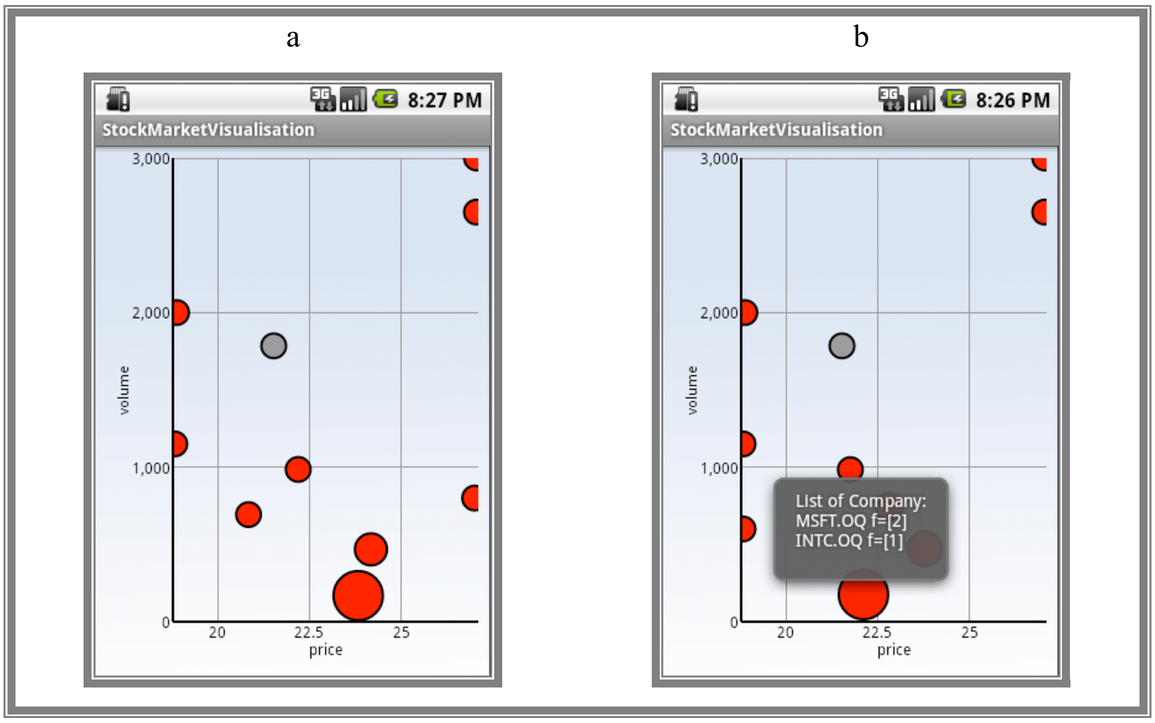


Figure 5.20 Selective Focusing interaction when visualising the stock market data in iCACV model

In conclusion, these interactive features allow users to accurately differentiate the visualised clusters. This shows the importance of interactivity in such systems by assuring that users are given the appropriate information from the visualised data according to the present context.

• Implication of Cluster Growth Control

This experiment aims to evaluate the effectiveness of having the Cluster Growth Controller as part of the iCACV framework. In this experiment, we use the real estate data source and we follow the same main scenario of the emergency call centre, where high-speed stream of data is being fed to the mobile phone to be analysed and visualised.

In order to assess the implication that the Cluster Growth Controller can bring to the UDM visualisation model, we evaluate the amount of clutter on screen and the speed at which the screen gets cluttered with the use of Cluster Growth Controller and without it. Indeed, the speed at which the screen gets cluttered can be a good indicator of better or poorer visualisation. For instance, in one experiment, the screen might get cluttered after visualising only 1000 data items, while in another experiment, the model might keep the screen uncluttered until 1500 items. This is regarded as improvement for the data visualisation when using the same dataset since it keeps the screen uncluttered for larger amount of time.

By comparing the results from both cases (with cluster growth control and without), we can find out whether incorporating such interaction technique can actually reduce the clutter or not. Like in the previous experiments, we measure clutter by continuously examining the percentage of the screen that is busy with data, which we refer into as Coverage Percentage. Also, clusters distinctiveness is another visual indicator of the amount of clutter on screen. Moreover, since iCACV (and CACV) has different visualisation modes, we can examine the status mode as an evaluation parameter that indicates when the system switches from one visualisation mode to the other. Knowing the time of switching between modes can help us learn how much clutter was there since this switching process is basically based on the amount of clutter on the screen.

In order to attain as appropriate and realistic results as possible, we run the experiment five times for each settings using the same dataset. This is to get more accurate results by averaging all the available results from all cases. We do the same thing for different coverage threshold values to examine if changing that value affects the effectiveness of the Cluster Growth Controller.

In measuring the effectiveness of the Cluster Growth (CG) control, we run similar testing cases for the iCACV with the cluster growth control threshold (Cluster Growth Threshold > 0) and without it (Cluster Growth Threshold = 0). The results are shown below in figures 5.21, 5.22 and 5.23.

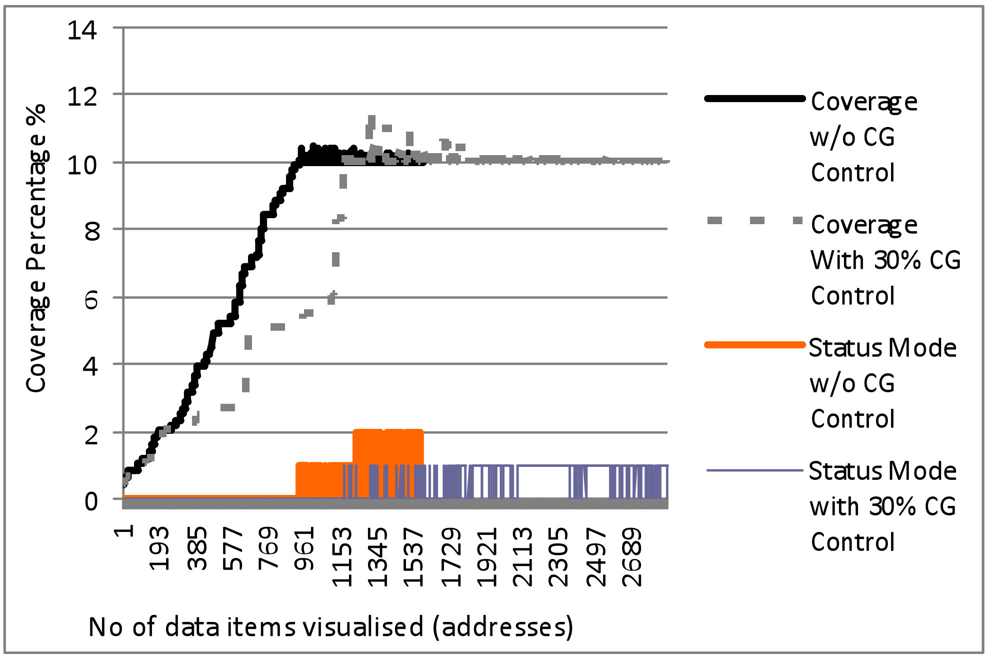


Figure 5.21 coverage percentage (clutter) with the implication of incorporating the Cluster Growth (CG) control, at 10% coverage threshold

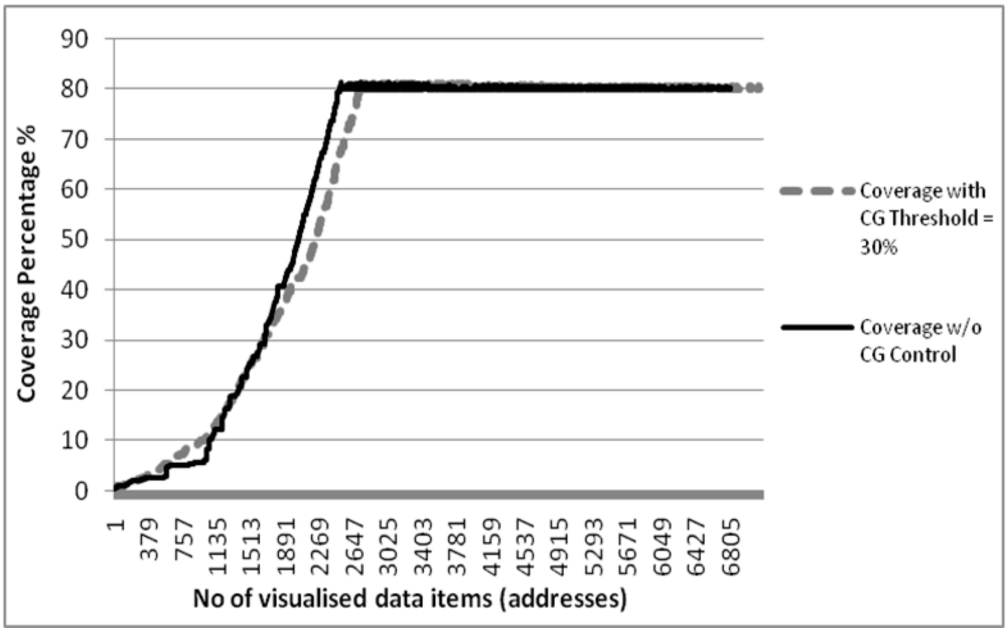


Figure 5.22 coverage percentage (clutter) with Cluster Growth Threshold equal to 30%

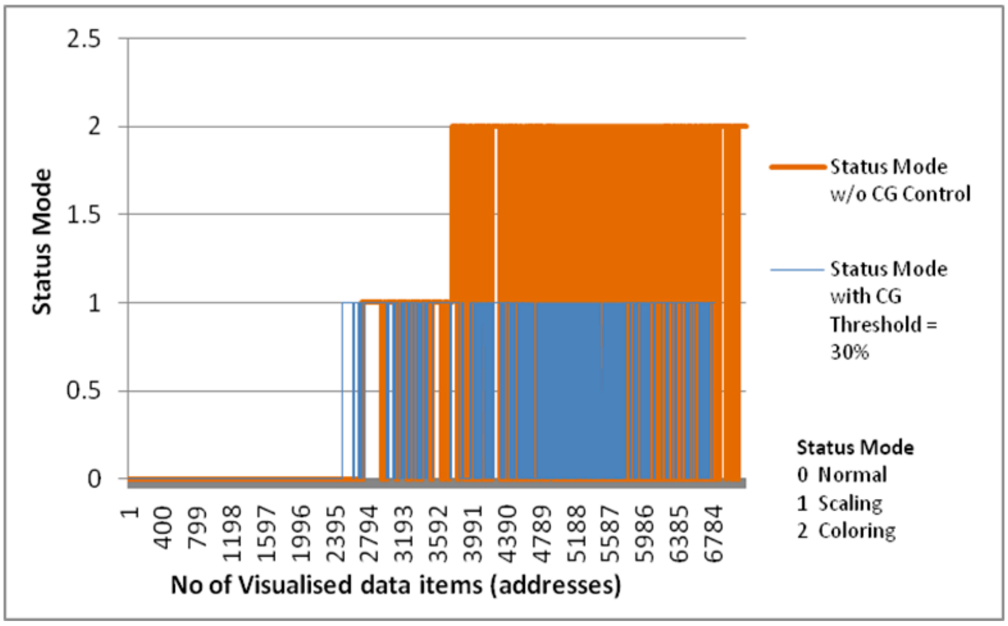


Figure 5.23 visualisation mode of both cases (with and without Cluster Growth threshold at 30%) coverage threshold = 80%

Figure 5.21 shows that using the Cluster Growth Control has reduced the coverage (clutter) on screen significantly. Also, from looking at the status (visualisation mode) in figure 5.23, the iCACV switches to the colouring mode (status mode = 2) after visualising around 3833 objects, whereas the iCACV algorithm did not switch to colouring when using the Cluster Growth Controller even after visualising more than 6707 objects. In other words, in the second case when Cluster Growth Controller was used, the amount of clutter did not even reach the threshold (coverage threshold) to switch to different mode. In summary, the figures show that incorporating Cluster Growth control can reduce clutter by more than 54%, which in turn can enhance the overall visualisation of the UDM outcomes.

• Effectiveness of Providing Audio Feedback for Off-Screen objects

Our iCACV model uses synthesis speech (voice) to give feedback to the user about the off-screen objects. While this keeps users always informed quickly and accurately, it maintains the clutter to the lowest level by not adding any sign or affecting the visualisation process. This could be especially important for users who are moving in the field, but they still need to stay updated instantly and accurately as in the case of the fire fighters in our case scenario. Also, in some cases, when the user has a little amount of charge in his/her mobile phone and he/she still wants to stay updated, the audio feedback can be critically useful. Audio feedback can be used to inform users when any cluster is growing into an alarming size to get their attention without a need to keep the screen ON.

This not only ensures the reduction of energy usage, but it also provides different way of communicating with the user. Users are human beings, and so they have limited amount of resources that they can assign to different tasks at once, as Campbell and Tarasewich emphasised in their article (Campbell and Tarasewich, 2004). Therefore, having the Audio Feedback Handler as part of the iCACV framework enables users to switch to the appropriate type of alert (audio or visual) based on their preference and their situation. A Simple demonstration video has been put available online for more explanation as follows:

http://www.youtube.com/watch?v=g7rlgqEIi7o.

• Effectiveness of Screen Fencing of Clusters

Theoretically, having the UDM visualisation algorithm to visualise only the on-screen objects using the Screen Fencing technique should reduce the computational overhead, decrease the energy consumption, and enhance the resource management as explained earlier in chapter 3. This concept was also emphasised by Karstens et al. (Karstens, et al. 2003), where they showed that limiting the number of visualised items could improve performance and enhance the resource management. Practically, However, our experimental assessment of the Screen Fencing of clusters on Google Android did not show remarkable improvements as it can be seen in figure 5.24. We have executed the same scenario as in the previous evaluation cases for five times to get the average of all the testing cases. We aimed to measure the energy consumption over time when implementing Screen Fencing and when we do not have it, which is basically the case when comparing CACV and iCACV. The difference in the results was very minor and did not show much improvement as expected.

The main explanation is that Google Android mobile phone is able to take care of this issue and that Android OS does the filtering of the off-screen objects itself regardless of wether the application is filtering them or not. This means that the Screen Fencing technique that we are aiming to implement in the iCACV is already exist in the underlying libraries of Android OS and that it already does the off-screen objects filtering itself. To prove our explanation, the testing experiment needs to be implemented on another mobile platform to find out how much the Screen Fencing of clutter can improve performance, reduce energy consumption, and enhance the resource management. However, this is outside the scope of this dissertation and is planned for future work.

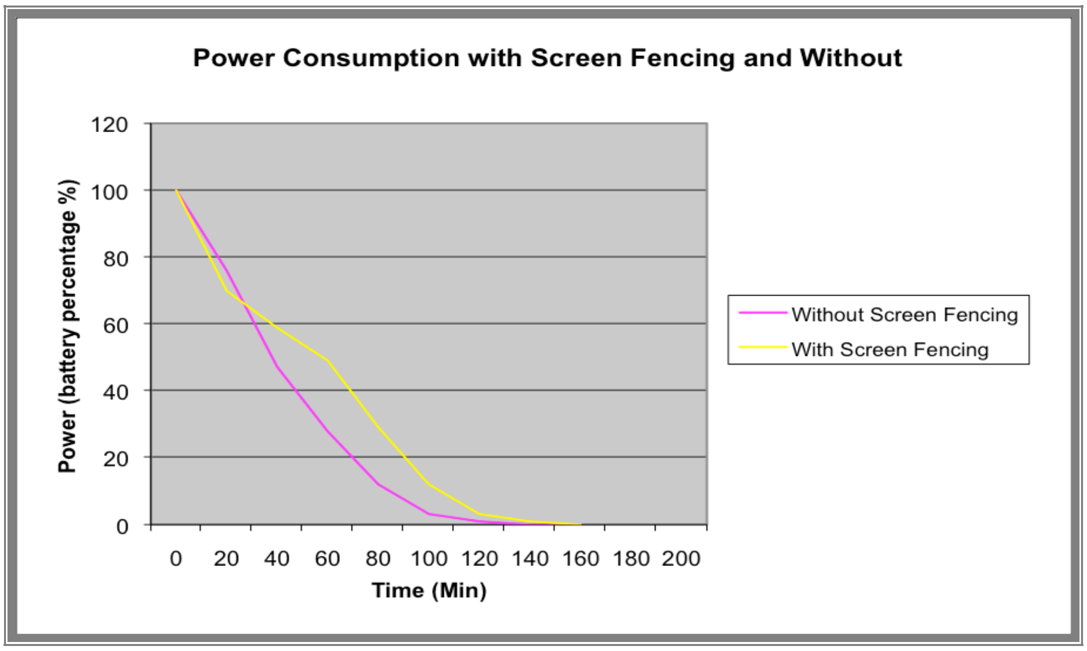


Figure 5.24 power consumption vs. time when incorporating screen fencing of clusters

5.4 Conclusions

In summary, this chapter has presented the iCACV framework in a generic multi-layer format, and then we explained each component of its architecture. Our improvements can be seen as a layer of interactive clutter awareness on top of the existing CACV algorithm that already has the resource awareness and the energy awareness. As a result, iCACV combines the strengths of CACV (resource and energy awareness) and the clutter reduction techniques through interaction.

iCACV consists of four main modules that attempts to improve data visualisation and resource management through interaction. Those modules are: User Preference

Monitor, Audio Feedback Handler, Cluster Growth Controller, and the Screen Fencing Controller. First, the User Preference Monitor aims to improve the user's comprehension of data by listening to any user's input and it also enables updating/personalising the data visualisation process according to the user's desire. The Audio Feedback Handler, on the other hand, is responsible for enhancing the resource management and the data visualisation by delivering appropriate audio alert when clusters are growing alarmingly, and if the user desired.

Moreover, The Cluster Growth Controller is the second main module in iCACV framework, which is responsible for controlling the growth of clusters to ensure better understanding of clusters at the least possible clutter. Furthermore, the Screen Fencing Controller represents an elegant approach of software engineering that can reduce the unnecessary overhead that might be produced by visualising the off-screen objects.

Furthermore, in this chapter, we have presented the implementation and the main components of the iCACV model. We first presented the software development environment that was used, followed by code samples, discussion, and detailed explanations of each module of the iCACV framework.

In terms of evaluation, we have presented a case scenario, where an emergency call centre is streaming the addresses of the callers to be analysed and visualised on mobile phones to help officers to make well-informed decision in such cases. We presented the details of our experimental system by explaining the main components and the datasets that we used.

Finally, we have shown through figures and screenshots that incorporating interaction to the UDM visualisation model can improve the data visualisation on those devices and can also enhance the resource management. Large number of experiments was conducted as explained in each case to guarantee as much appropriate results as possible. Our findings were clearly shown in the figures that we presented earlier to indicate how much could the interaction capabilities improve the UDM visualisation process in terms of reducing clutter, improving visualisation, and promoting better resource management.

6 Conclusions

6.1 Research Summary and Contribution

Mobile data stream mining visualisation is an emerging area of research that has attracted many researchers recently due to the recent growth in mobile phone capabilities and the advances of data acquisition systems. UDM visualisation involves analysing a continuous flow of data and then visualising the outcome of this analysis in time critical scenarios on mobile devices. This helps mobile users to have access to their intelligent systems to make well-informed decisions anywhere at anytime. However, visualising such unpredictable, massive, and continuous data streams on mobile devices brings many challenges. Examples of such challenges are constrained resources, limited screen space, limited battery supply, mobility, and connectivity issues. This demands a highly adaptive visualisation strategy that can cope with all these challenges and efficiently and accurately perform the UDM visualisation on smart phones. Our literature review has revealed no strategy that is adaptive, interactive, and scalable, that can be up to the task. Indeed, many researchers have mentioned that such a strategy is highly needed and it does not exist. Thus, this thesis aimed to propose, develop, and evaluate a generic, interactive, and adaptive UDM visualisation strategy that can reduce clutter, improve data visualisation, and promote better resource management.

In this thesis, we have reviewed the literature comprehensively to learn, explore, and investigate the UDM visualisation concepts, techniques, and challenges. We started by looking at data stream mining and data visualisation in general. We then moved our focus to the UDM visualisation precisely. From that, we concluded a list of essentials that must be included in any UDM visualisation system to be successful. We then used this knowledge to propose our interactive clutter aware clustering visualiser model. In iCACV, we aimed to address the main concerns and challenges of UDM visualisation by different means of interaction, selection, and user-driven data presentation. The iCACV framework consists the following:

- User Preferences Monitor
- Audio Feedback Handler
- Cluster Growth Controller
- Screen Fencing Controller

We have implemented iCACV based on CACV to incorporate CACV strengths (resource and energy awareness) and the interactive clutter reduction techniques. In order to evaluate our model, we have designed and implemented a complete and functional experimental system that feeds different kind of datasets (addresses and stock market data) to be analysed and visualised on Google's Android Mobile phone. The systematic and statistical evaluation has shown better visualisation, less clutter, and enhanced resource management. In fact, in some experiment, we could reduce clutter by more than 54% as in the experiment of evaluating the Cluster Growth Controller. We have shown the effectiveness of our improvements by statistically drawing some graphs and/or by screenshots from the operational Google Android application. Moreover, some other capabilities of the iCACV have already been proven to be efficient before by some other researchers and we just adopted them from the mobile visualisation field. This includes Selective Focusing, and Screen Fencing. However, the Screen Fencing technique did not show remarkable cut in the use of energy, which can be reasoned to the possibility that Android OS already looks after filtering off-screen objects when visualising large spatial dataset.

6.2 Future Work

In this section, we aim to highlight the future direction of our research and the opportunities in this field. We have demonstrated the importance of interaction in UDM visualisation. We have also seen its impact on improving visualisation and data perception. However, few issues need to be addressed in this regard. First, our results for the Screen Fencing technique did not show remarkable improvements on the use of energy, which was not expected. This issue needs to be investigated further. Second, in our list of the UDM essentials, which we built based on our comprehensive literature review, other factors are very important to the UDM visualisation. This includes human factors and evaluation. Many experts from the human computer interaction field emphasised on the importance of having knowledge in the human factors that might affect the data visualisation and perception. Also, there were some claims about the importance of evaluating the system on real users. Thus, these aspects need to be studied well in order to address some of the common UDM visualisation challenges. Finally, given the impressive results that we attained from using solely basic means of interaction, we believe that bringing the concept of “augmented reality” to the mobile context, will improve the performance, and the data visualisation further. Therefore, this will be focus of our future research.

References:

Aggarwal, C. C. (2006). “Data Streams: Models and Algorithms.” Springer, USA.

Aggarwal, C. C. Han, J. Yu, P.S. (2004). “On Demand Classification of Data Streams.” *The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* p. 503-508, Seattle, WA, USA, August 2004.

Avnur, R. and Hellerstein, J. (2000). “Eddies: Continuously adaptive query processing”. *In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, p. 261–272, Dallas, Texas, USA, May 2000.

Babcock, B. Babu, S. Datar, M. Motwani, R. and Widom, J. (2002). “Models and issues in data stream systems”. *In Proceedings of Symposium on Principles of Database Systems, PODS*, Madison, WI, USA, June 2002.

Barbara', D. (2002). "Requirements of clustering data streams." *ACM SIGKDD Explorations* **3**(2): 23-27.

Bertin, J. (1981). Graphics and Graphic Information-Processing. Walter de Gruyter, New York, USA.

Björk, S. Holmquist, L.E. Redstrom, J. Bretan, I. Danielsson, R. Karlgren, J. Franzen, K. (1999). “WEST: a Web browser for small terminals”, *Proceedings of the 12th annual ACM symposium on User interface software and technology*, p. 187-196, New York, NY, USA, 1999. ACM.

Burigat, S. and Chittaro, L. (2005). “Visualising the results of interactive queries for geographic data on mobile devices” *Proceedings of the 13th annual ACM international workshop on Geographic information systems,* p. 277-284, Bremen, Germany, October 2005, ACM.

Burigat, S. Chittaro, L. et al. (2006). “Visualizing locations of off-screen objects on mobile devices: a comparative evaluation of three approaches.” *Proceedings of the 8th* *conference on Human-computer interaction with mobile devices and services,* p. 239-246, Espoo, Finland, September 2006, ACM.

Burigat, S. Chittaro, L. Parlato, E. (2008). “Map, diagram, and web page navigation on mobile devices: the effectiveness of zoomable user interfaces with overviews.” *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services,* p. 147-156, Lisbon, Portugal, September 2008, ACM.

Buyukkokten, O. Garcia-Molina, H. Paepcke, A. and Winograd, T. (1999) “Power Browser: Efficient Web Browsing for PDAs. Technical Report. Stanford InfoLab.” (*Publication Note: Human Factors in Computing Systems Conference (CHI 2000)*, The Hague, The Netherlands), April 2000.

Campbell, C. and Tarasewich, P. (2004) “What Can You Say with Only Three Pixels?” *Mobile Human-Computer Interaction – MOBILEHCI 2004*, vol 3160, p. 15-35. Springer, Berlin, Heidelberg, September 2004.

Card, K.S. Mackinlay, J. Schneiderman, B. (1999). “Readings in information visualization: Using vision to think.” *1999 MORGAN Kaufmann publisher Inc,* p. 579 – 581, San Francisco, California, October 1999.

Charaniya, A. and Lodha, S. (2003). “Speech Interface for Geo-Spatial Visualization.” Proceedings of the IASTED International Conference on Modelling, Simulation and Optimization - MSO 2003, p. 394-148, Banff, Canada, July 2003.

Chittaro, L. (2006). "Visualizing information on mobile devices." *Computer* **39**(3): 40-45.

Domingos, P. and Hulten, G. (2001). “A General Method for Scaling Up Machine Learning Algorithms and Its Applications to Clustering.” in Proceedings of the 18th International Conference on Machine Learning, p. 106-113, Morgan Kaufmann, 2001.

Duffield, N. and Grossglauser, M. (2000). “Trajectory sampling for direct traffic observation”. *In Proceeding of the 2000 ACM SIGCOMM the annual conference of the Special Interest Group on Data Communication*, p. 271–284, Stockholm, Sweden, Sept. 2000

Dunlop, M. and Brewster, S. (2002), “The Challenge of Mobile Devices for Human Computer Interaction”, *Personal and Ubiquitous Computing archive*, Vol **6**(4), Springer-Verlag, London, UK, September 2002.

Ellis, G. and Dix, A (2007), “A Taxonomy of Clutter Reduction for Information Visualisation”, *Visualisation and Computer Graphics*, vol. **13**, no. 6, p. 1216-1223, http://www.ieeexplore.ieee.org/ accessed on 21 October 2009.

Encarnacao, J. M. Frühauf, et al. (1995). “Mobile Visualization: Challenges and Solution Concepts.” *In Proceeding of the Fifth International Conference on Computer Applications in Production and Engineering* (CAPE’95), Beijing, China, May 1995.

Enke, D. and Thawornwong, S. (2005), “The use of Data Mining and Neural Networks for Forecasting Stock Market Returns”, *Expert Systems with Applications*, vol. **29**, no. 4, p. 927-940.

Fuchs, G. and Schumann, H. (2006). “Visualization in Multimodal User Interfaces of Mobile Applications.” Information Resources Management Association Washington DC, USA.

Gaber, M. M. (2009), “Data Stream Mining Using Granularity-Based Approach”, *Studies in Computational Intelligence*, vol. **206**, p. 47-66, http://www.springerlink.com/content/n68g00w36r836333/ accessed on 21 October 2009.

Gaber, MM. Krishnasawamy, S. Gillick, B. Nicoloudis, N. Liono, J. Zaslavsky, A. (2010) “Adaptive Clutter-Aware Visualisation for Mobile Data Mining”. *Proceedings of the 22nd International Conference on Tools with Artificial Intelligence*, Arras, France, October 2010.

Gaber, M. Krishnaswamy, S. et al. (2003). "Adaptive Mining Techniques for Data Streams using Algorithm Output Granularity." *The Australasian Data Mining Workshop (AusDM 2003),* Held in conjunction with the 2003 Congress on Evolutionary Computation (CEC 2003), Canberra, Australia, Springer Verlag, Lecture Notes in Computer Science (LNCS), 2003.

Gaber, M.M. Krishnaswamy, S. and Zaslavsky, A. 2005. “On-board Mining of Data Streams in Sensor Networks”, *A Chapter in Advanced Methods of Knowledge*

Discovery from Complex Data, (Eds.) S. Badhyopadhyay, U. Maulik, L. Holder and D. Cook, Springer.

Gaber, M.M. Krishnaswamy, S. and Zaslavsky, A. (2004), “Cost-Efficient Mining Techniques for Data Streams”, *Proceedings of Australasian Workshop on Data Mining and Web Intelligence (DMWI2004)*, p. 109-114, Dunedin, New Zealand, Purvis, M., Ed. ACS.

Gaber, M.M. Krishnaswamy, S. and Zaslavsky, A. (2004), “Ubiquitous Data Stream Mining”, *Current Research and Future Directions Workshop Proceedings held in conjunction with The Eighth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, Sydney, Australia, 26 May 2004, http://www.csse.monash.edu.au/~mgaber/CameraReadyPAKDD.pdf accessed on 21 October 2009.

Gaber, M. and Yu, P. (2006). "A holistic approach for resource-aware adaptive data stream mining." New Generation Computing **25**(1): 95-115.

Gaber, M.M. Zaslavsky, A. and Krishnaswamy, S. (2004), “Resource-Aware Knowledge Discovery in Data Streams”, *Proceedings of First International Workshop on Knowledge Discovery in Data Streams*, held in conjunction ECML and PKDD 2004, Italy, pp. 649-656, September 2004, http://www.lsi.us.es/~aguilar/ecml2004/FP4.PDF accessed on 21 October 2009.

Gaber, M.M. Zaslavsky, A. and Krishnaswamy, S. (2004), “Towards an Adaptive Approach for Mining Data Streams in Resource Constrained Environments”, *Data Warehousing and Knowledge Discovery*, *Lecture Notes in Computer Science*, vol. **3181**, pp. 189-198, http://www.springerlink.com/content/5ky3vje6fcyr9cw8/ accessed on 21 October 2009.

Gaber, M.M. Zaslavsky, A. and Krishnaswamy, S. (2005), “Mining Data Streams: A Review”, *ACM SIGMOD Record*, vol. **34**, no. 2, p. 18-26.

Giannella, C. Han, J. Pei, J. Yan, X. *and* Yu, P.S. (2003). “Mining Frequent Patterns in Data Streams at Multiple Time Granularities.” *Next Generation Data Mining*, AAAI/MIT, MIT Press.

Gillick, B. Krishnaswamy, S. et al. (2006). "Visualisation of fuzzy classification of data elements in Ubiquitous data stream mining." IWUC **6**: p. 29-38.

Golab, L. and Ozsu, M.T. (2003). "Issues in Data Stream Management." SIGMOD Record **32**(2): p. 5-14.

GSM Arena, http://img.gsmarena.com/vv/pics/htc/htc-dream-3.jpg accessed on 9 October 2009.

Guha, S. Mishra, N. Motwani, R. O'Callaghan, L. (2000). "Clustering Data Streams." *Annual Symposium on Foundations of Computer Science*, p. 359, Redondo Beach, California, November 2000, IEEE.

Gustafson, S. Baudisch, P. Gutwin, C. and Irani, P. (2008) "Wedge: Clutter-Free Visualization of Off-Screen Locations." in *Proceedings of CHI 2008,* p. 787-796, *Florence, Italy*, April 2008.

Han, J. and Kamber, M. (2006), "Data Mining: concepts and techniques", edn. 2, Morgan Kaufmann.

Hand, D.J. (1999), "Statistics and Data Mining: Intersecting Disciplines", *ACM SIGKDD Explorations*, vol. **1**, no. 1, p. 16-19.

Holland, S. and Morse, D.R. (2001) "Audio GPS: Spatial Audio in a Minimal Attention Interface." *In Proceedings of Mobile HCI 2001*, Lille, France.

HTC Corp (2009), "HTC Dream Specification", viewed October 5 October 2009, http://www.htc.com/www/product/dream/specification.html.

Johnson, P. (1998), "Usability and Mobility; Interactions on the Move." *In Proceedings of the 1st Workshop on HCI for Mobile Devices*, University of Glasgow, UK 1998.

Kargupta, H. Bhargava, R. Liu, K. Powers, M. Blair, P. Bushra, S. Dull, J. Sarkar, K. Klein, M. Vasa, M. and Handy, D. (2004). "VEDAS: A Mobile and Distributed Data Stream Mining System for Real-time Vehicle Monitoring" *in Proceedings of 2004 SIAM International Conference on Data Mining (SDM'04)*, Lake Buena Vista, FL, April 2004.

Kargupta, H. Park, B. Pittie, S. Liu, Li. Kushraj, D. Sarkar, K. (2002). "MobiMine: Monitoring the stock market from a PDA." ACM SIGKDD Explorations Newsletter **3**(2): 37-46.

Karstens, B. Kreuseler, M. Schumann, H. (2003). "Visualization of complex structures on mobile handhelds". *In Proceedings of IMC'2003, International Workshop on Mobile Computing*, p. 8, Rostock, Germany, (2003)

Kjeldskov, J. and Graham C. (2003). "A Review of Mobile HCI Research Methods", *Proceedings of the 5th International Mobile HCI 2003 conference*, p. 317-335, Springer-Verlag, Udine, Italy, September 2003.

Kjeldskov, J. Skov, M.B. Als, B.S. and Høegh, R.T. (2004), "Is it Worth the Hassle? Exploring the Added Value of Evaluating the Usability of Context-Aware Mobile Systems in the Field." *In Proceedings of the 6th International Mobile HCI 2004 conference.* LNCS, Springer-Verlag.

Krishnaswamy, S. Gaber, M.M. Harbach, M. Hugues, C. Sinha, A. Gillick, B. Haghighi, P.D. and Zaslavsky, A. (2009), "Open Mobile Miner: A Toolkit for Mobile Data Stream Mining", *Knowledge Discovery and Data Mining 2009*, Paris, 2009, http://www-ai.cs.uni-dortmund.de/PROCEEDINGS/SIKDD2009/demos/D02-kdd09demo.pdf accessed on 21 October 2009

Krishnaswamy, S. Gaber, M.M. Nicoloudis, N. Zaslavsky, A. and Gillick, B. (2009), "Visualisation of Real-Time Data Analysis for Mobile Devices", Australian Provisional Patent Application: 2009903925.

McCrickard, D.S. Catrambone, R. Chewar, C.M. and Stasko, J.T. (2003) "Establishing Tradeoffs that Leverage Attention for Utility: Empirically Evaluating Information Display in Notification Systems." *International Journal of Human-Computer Studies,* vol. **8**, p. 547-582, 2003.

Liono, J. (2009). "Energy Efficiency of Ubiquitous Data Mining Visualisation." Caulfield School of IT - DSSE. Melbourne, Monash University. **Master Thesis**: 84.

Nicholson, M. and Vickers, P. (2004), "Pen-Based Gestures: An Approach to Reducing Screen Clutter in Mobile Computing", *Mobile Human-Computer Interaction*, *Lecture*

Notes in Computer Science, Springer Berlin, vol. **3160**, pp. 320-324, http://www.springerlink.com/content/6frruv84l7l2e0h4/ accessed on 21 October 2009.

Ordonez, C. (2003). "Clustering Binary Data Streams with K-means." ACM, DMKD **03,** p. 12-20, June 2003.

Paelke, V. Reimann, C. Rosenbach W. (2003) "A Visualization Design Repository for Mobile Devices." *In proceedings of the 2nd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, p. 57-62, 2003.

Peng, W. Ward, M.O. and Rundensteiner, A. (2004), "Clutter Reduction in Multi-Dimensional Data Visualization Using Dimension Reordering", *Information Visualization*, Austin, Texas, pp. 89-96, 2004, http://www.ieeexplore.ieee.org/ accessed on 4 April 2009.

PhoneArea (2009), http://phonearea.net/files/2009/03/android-dev-phone-1.jpg accessed on 9 October 2009.

Rodrigues, J. Goni, A. and Illarramendi, A. (2005). "Real-Time Classification of ECGs on a PDA." IEEE T. on IT in Biomedicine **9**(1): 23-24.

Savio, N. and Braiterman, J. (2007) "Design Sketch: The Context of Mobile Interaction". *Mobile HCI 2007*, Singapore, September 2007.

Shah, R. Krishnaswamy, S. Gaber, M.M. (2005). "Resource-Aware Very Fast K-Means for Ubiquitous Data Stream Mining." *The 2nd International Workshop on Knowledge Discovery in Data Streams, in conjunction with the 16th European Conference on Machine learning (ECML 2005) and the 9th European Conference on the principals and Practice of knowledge Discovery in Databases*, Porto, Portugal, (PKDD) 2005.

Spence, R. (2006). "Information visualization", 2nd Edition, ISBN 0132065509, Pearson Education, USA.

Tonder, B.V. and Wesson, J. (2008). "Using adaptive interfaces to improve mobile map-based visualisation." *In Proceedings of the 2008 annual research conference of the South African Institute of Computer Scientists and Information Technologists on IT research in developing countries* (SAICSIT '08), p. 257-266, Garden Route, Wilderness, South Africa, October 2008, ACM.

Trevor, J. Hilbert, D.M. Schilit, B.N. and Koh, T.K. (2001): "From desktop to phonetop: a UI for web interaction on very small devices". *Proceedings of the 14th annual ACM symposium on user interface software and technology (UIST2001)*, Orlando, FL, USA, November 2001.

Wang, H. Fan, W. Yu, P.S. Han, J. (2003). “Mining Concept-Drifting Data Streams using Ensemble Classifiers”. *The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, Washington DC, USA, ACM.