
INF-3981
Master thesis in
Computer Science

# Improving the usefulness of multimedia archives by applying data reduction techniques

by

**Tjalve Aarflot**

February 5th, 2009

Faculty of Science
**Department of Computer Science**
**University of Tromsø**

# Abstract

In recent years, an increasing amount of personal images, video, sound and text data are captured and stored in a digital format. Increased storage capacity at lower cost entice us to attempt to store everything, but without effective information retrieval techniques, the usefulness of the data becomes limited.

Some people have taken personal data capture to extremes and have begun to capture digitally all aspects of their life, which creates enormous archives of multimedia data. This is not a new idea: in 1945 Vannevar Bush wrote his visionary article "As We May Think" where he described Memex, the first Human Digital Memory (HDM). Today we have projects like Microsoft MyLifeBits building on the Memex vision, however there has been little focus on organizing this kind of data effectively.

By applying data reduction, we show the benefits of removing redundancy from HDMs, and illustrate how the same data reduction framework can be used to effectively support information access from HDMs.

# Acknowledgments

The author would like to thank his supervisor, Professor Dag Johansen, and co-supervisor, Dr. Cathal Gurrin, for valuable ideas, support and motivation.

A special thanks goes to Dr. Gurrin for allowing usage of his personal Lifelog dataset, and for help in evaluating the system.

Thanks to Dr. Håvard Johansen for valuable feedback on document structure.

Thanks to the CDVP team at DCU for information and guidance during the stay in Dublin, and for providing all the metadata they have collected in previous works.

Finally, thanks to the rest of the iAD team at UiT for all helpful discussions.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Background

Naturally following the ongoing trend of increased proliferation of personal computational devices with audio, picture and video capturing capabilities, more and more media content are now available in digital rather than analogue formats. This spawns both new possibilities and challenges for interacting with the media.

No longer limited only by textual descriptions of knowledge and experiences, huge digital libraries can now store both personal and business data in a representation much closer to how we perceive them, opening up for finally fulfilling Vanevar Bush' vision from 1945 of a Human Digital Memory (HDM) [12]. Everyday we add events and interactions to our memories, but they are known to become more unreliable as we age, and we can not share the directly with our fellow human beings. Lifelogging is part of a vision where we will soon be able to digitally record all we see and hear into HDMs. As this technology improves, HDMs are believed to make a huge impact both socially, personally and medically in the future.

One of the goals of digital libraries used in HDM and Lifelogging systems, is to store everything, always, so that we in 70 years can have a perfect digital memory from our life available. However, shown in the IDC report of 2007 [1], we are now creating more information than we are creating storage space. This means that even with the rapid drop in storage media prices the recent years, we will no longer manage to store everything. Additional challenges arise from the fact that information retrieval from vast multimedia archives is hard to do well, caused by a combination of information overload and the semantic gap.

Data reduction by compression is already utilized by most popular media formats in use today, and simple algorithms for deletion of oldest or random data cannot yield satisfying results if our goal is to preserve information. A more

advanced approach is needed, where the stored digital objects can be automatically analyzed and understood so that smart decisions on how to reduce their data payload can be made, without deleting all their information. In [3] we introduced a framework supporting such transient digital objects in digital libraries.

Focus in this thesis will be to expand on that work and move our techniques into the more advanced domain of life logging. The data will now be much more diverse than in the video surveillance domain used in our previous system, and the semantic information in the data will be more dynamic and highly personal to the owner of the data.

## 1.2 Problem definition

*This thesis shall develop and evaluate in a scientific context a framework and prototype system for automatic reduction of data through filtering techniques. The specific application domain at study is related to the Lifelog project. The thesis shall focus on how to support a system enabling accurate data reduction, but retaining data quality.*

## 1.3 Interpretation

Lifelogging automatically captures images rich in metadata from the viewpoint of the wearer throughout the day, and creates a visual history log of a person's day (amounting to roughly one million images over the course of a year). This thesis use Lifelog data collected by the CDVP team at DCU.

Our previous prototype framework for video surveillance [3] demonstrated the potential in automatically reducing the data set (without losing too much information). That framework will now be extended and a new prototype will be developed to apply and evaluate the extended framework on Lifelog input data.

We will point to some shortcomings in the previous model, and show that for a more advanced set of data such as this, the framework needs to be extended to allow for query-time filtering. To do data reduction on the Lifelog domain we will develop a quality filter and an image similarity filter. These filtering algorithms can be applied to the complete dataset, but also at query-time to improve presentation and reduce the amount of data transferred to the user. This will improve the usefulness of the Lifelog system.

Typical metadata collected or generated for the Lifelog domain can include, for instance, location, lighting conditions, temperature, and situation present on the images captured. We will use this to transform the objects so they are reduced to contain the most unique and important data based on both the explicit query terms and implicit query-time information such as the display capability of the user's interface. To do this, the previous model must be extended to also allow for dynamic filters that are applied only at presentation time data, in addition to global static filters that work over the complete dataset.

## 1.4 Method

The discipline of computing is divided into three paradigms [5]. These are *theory*, *abstraction* and *design*.

*Theory* is rooted in the field of mathematics, and consists of four steps:

1. Characterize the objects under study.
2. Make hypothesis about possible relationships among them.
3. Decide whether the hypothesis are true or false.
4. Interpret the results.

With the mathematical approach, one would expect to repeat these steps each time some inconsistencies or errors are found.

*Abstraction* is rooted in experimental, scientific methods, and consists of the following steps to investigate a phenomenon:

1. Find a hypothesis for the observed phenomenon.
2. Construct a model and make a prediction.
3. Design an experiment and collect data.
4. Analyze the results.

The scientist expects to repeat these steps when the model's predictions disagree with the analyzed results.

This thesis will follow the *design* paradigm, which is rooted in engineering and consists of the following steps:

1. State requirements for the system.
2. State specifications for the system.
3. Design and implement the system.
4. Test and evaluate the system.

This means that a system solving a specific problem, reflected in the requirement specification, will be developed. A solution that fulfills these requirements will be designed and implemented. Finally, the system will be tested and evaluated by comparing the test results with the requirement specification.

## 1.5 Project Context

Technology for accessing information has made a huge impact in a wide range of industries, business models and even social patterns, and has emerged as one of the most innovative technology areas recently. The market for information access and enterprise search has already been revitalized, with users now able to access more information across an expanding set of content stores, including e-mail, file servers, intranets, extranets and the Web. New innovation will take these technologies and create new markets in the years to come.

The iAD (information Access Disruption) Center[1] targets core research for next generation precision, analytics and scale in the information access domain. Partially funded by the Research Council of Norway as a Center for Research-based Innovation (SFI), iAD is directed by Fast Search & Transfer[2], a Microsoft subsidiary, in collaboration with Schbisted[3], Accenture, and Cornell University, University College Dublin, Dublin City University, BI Norwegian School of Management and the universities in Troms (UiT), Trondheim (NTNU) and Oslo (UiO).

Focus for the iAD project is large-scale future-generation information access applications. The project will investigate structuring techniques and fundamental research issues like, for instance, how to best partition an application into a set of cooperating modules, how to optimize interaction among them, how and where to deploy them, how to interact with the users, how to provide integrity, security and auditing, and how to ensure fault-tolerance.

In particular, iAD is targeting next-generation search solution, which will have to deal with both a huge increase in the amount of data to handle, and new and emerging media types such as video and audio content in addition to sensor data. Another aspect is real-time situations where the value of data elements are determined by temporal relations in the data. To cope with this, next-generation information access system must be able to work together with systems where data-delivery, querying and processing are done over real-time data-streams.

## 1.6 Outline

This thesis consists of the following chapters:

**Chapter 2 - Related work** introduces the domain of Lifelogging and present some current work.

**Chapter 3 - Requirements** states the system requirements.

**Chapter 4 - Design** proposes a design for a framework based on the requirements

**Chapter 5 - Prototype Implementation** describes our prototype implementation of this framework for the Lifelog domain.

[1] www.iad-centre.no
[2] www.fast.no
[3] www.schibsted.com

**Chapter 6 - Experiments and Evaluation** describes the experiments, and presents and evaluates the results

**Chapter 7 - Conclusion** draws the conclusion of this thesis and propose possible future work.

# Chapter 2

# Related Work

## 2.1 The Lifelog domain

Everyday we add events and interactions to our memories, but they are known to become more unreliable as we age, and can not directly be shared with our fellow human beings. Lifelogging is part of a vision where we will soon be able to digitally record all we see and hear, into what is called a Human Digital Memory (HDM). As this technology improves, HDMs are believed to make a huge impact both socially, personally and medically in the future.

The concept was proposed already in MEMEX by Vannevar Bush in 1945 [12], and restated by Bill Gates in "The Road Ahead" (1995) [13]. Wearable digital systems that could perform passive image capture, thereby automatically saving our "memories", date at least back to work at MIT in the 1980's. In the next section we will look at some of the more recent work in the Lifelog domain.

## 2.2 Recent work

### 2.2.1 MyLifeBits

Microsoft's MyLifeBits [4] is perhaps the best known Lifelog project, where many aspects of Gordon Bell's life is digitally captured and organized. The system aims to avoid typical hierarchical media organization, instead offering good visualization of user's life bits as a main interface. First focusing on core features of a system for support HDM, it was later expanded to allow for more forms of capture such as radio and TV, and more recently to support SenseCam input [6]. The latter discuss the use of SenseCam data and metadata for automatic annotation of data to make it easier to browser for and find photos, for example by the use of GPS sensory input in combination with a

map interface. True to Bush's original vision, MyLifeBits can also capture IM[1] chat logs, telephone calls, web browsing history, and more. In short, the goal is to digitally duplicate and archive your human memory, at the core of most Lifelog systems. In the original paper from 2002, they find the vision from Memex on unlimited storage plausible, and state that it should be feasible for any user to simply upgrade their storage capacity as necessary. However, in [6] one of the issues discussed for their system is the requirement for quick lookup of items, an issue that surely do not become any easier in an unlimited-storage system.

### 2.2.2 Hori & Aizawa

In [11], Hori and Aizawa developed a system for capturing life-log video from a wearable camera, along with additional metadata from sensors such as a microphone, GPS receiver, accelerometer, gyro-sensor and a brain-wave analyzer. They estimate that to capture TV phone quality video for one person for a lifetime, only 11 TB of disk storage is necessary, and finds it very feasible. They argue that by using the additional sensory input they are able to process the video data at low computational cost. They also show however that broadcasting quality video for a lifetime would take up 736 TB which is a significant amount of data to store on a personal level, at least by today's standards. In general, their focus is not on storage capacity or processing quantities.

### 2.2.3 LifeStreams

More low-level than the later systems, LifeStreams from 1996 [14] was an early proposal for a way to store personal data outside of the bounds of traditional file systems, but at the time meant to contain everything a person deals with in his/her electronic life. More concerned with a computer desktop interface and file systems than being a true HDM, Freeman and Gelernter do make a number of arguments that are highly valid also today. For instance. they want archiving to be automatic, which corresponds to the passive capture systems used in [11, 6], and included ideas for automatic phone call record archives. Further they argue for sophisticated grouping of related objects, and to use a time-ordered view of the data, an idea more recently adopted by for instance Google's Picasa photo album software[2]. Maybe most interesting in relation to this thesis are their adoption of automatic summation and compression of the personal data, seemingly more concerned by storage space than post-2000 works in the area.

### 2.2.4 EyeTap

Another project worth mentioning is Steve Mann's EyeTap[3], which started with the "Wearable Wireless Webcam" idea in [15]. Improving on methods for capture since 1994, EyeTap is probably the most unobtrusive device for capturing

[1] Instant Messaging
[2] picasa.google.com
[3] www.eyetap.org

visual lifelog data today. A combination of the functionality from SenseCam and EyeTap seems like a natural next step. However, although technologically advanced, the focus is mostly on capturing alone, and does not consider management or access in huge HDM systems.

## 2.3 Lifelog issues

LifeStreams propose a storage system for personal data featuring automatic compression and summation. However, it was not made to handle today's vast collections of personal multimedia data, and more importantly does not contain methodology for analysis of such data, which is necessary to achieve compression outside that already built into modern media formats. In MyLifeBits and the work by Hori and Aizawa, no concerns are raised for storage space limitations, rather the view is that more storage space can always be added when necessary.

As pointed to by the IDF report from 2007 however, the dream of unlimited storage space where all information can be stored digitally seems to be a lost cause, with the total amount of information created in the world already surpassing the amount of new storage space produced [1]. Even if we had unlimited storage, that does not reduce the task of information retrieval in such systems. We could assume that computational power and information retrieval techniques will always improve as rapidly as the creation of new digital information, in which case the problem of information overload may be held at bay. In our previous work we instead proposed a framework for Transient Digital Objects in Digital Library systems, where we abandoned the "store everything" ideal and replaced it with smart filtering of digital objects to automatically reduce them while retaining information.

The framework was tested in a simple scenario of video surveillance with only one CCTV device monitoring a computer lab environment. The simple input data enabled us to use the framework to achieve very high data reduction by filtering for quality and redundancy.

In the next chapter we will discuss and present the necessary requirements of our framework applied to the domain of Lifelog data.

# Chapter 3

# Requirements

This chapter will present the requirements for our system. As discussed in the last chapter, we will expand our previous framework so it is more adjusted to address some of the issues in the domain of Lifelogging we find unanswered in other work.

The previous framework only considered transforming objects until they reached a minimal representation [6]. This approach gave good results in our experiments with CCTV video data, since simple filters could be applied to the data with a high accuracy of detecting redundant data. The Lifelog dataset however contains much more diverse data, since the SenseCam has captured any activity of the camera wearer, at most times of the day. The environment and context in the image data are unknown, and the high reduction achieved for video surveillance data in our previous prototype can no longer be expected.

Some data reduction can still be done without any extra information. Low quality images can be detected and removed from the objects, and we will use metadata from previous quality measurement work on this dataset [9] as the basis for a quality transformation rule. Further, we will develop an algorithm for detecting image similarity, so redundant images can be removed. Since a Lifelog system works as a Human Digital Memory personal to the owner of the data, deciding what is the most valuable image data in an object can really only be done by that person. Uninformed choices made without user input must be required to have a very high accuracy, so that only data that can be generally acknowledged to have low quality or be of high redundancy are removed.

Another issue not discussed in our previous framework proposal is how to support multi-modal access. The average amount of images in each object in this dataset was calculated to be 83. If access to the Lifelog system is made from a mobile device or another small-screen device, viewing 83 images on the screen at the same time makes for a bad user experience. On a mobile device maybe only four images will fit on the screen, so a lot of scrolling and navigation have

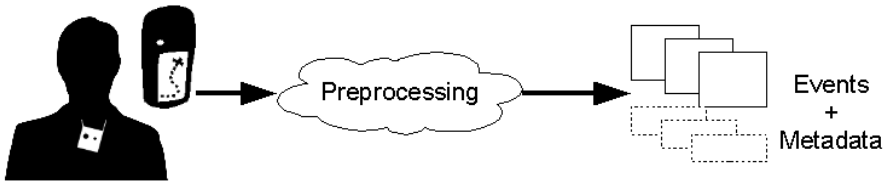



Figure 3.1: Preprocessing of raw input data into Events + Metadata.

to be performed to get a good view of the whole event captured. This will make it harder to use the view as a memory trigger, and scrolling and navigation on a mobile device is in any case cumbersome. Additionally, the data payload of 83 images is still very high to be sent to a mobile device as of 2009. A standard ranking of the images based on some available metadata and image metrics can be performed to send only the top 4 ranked images to the mobile device. But will this be the best representation of the object?

We want to do better, and therefore introduces query-time transformations to our digital library framework, which can transform objects until they reach a content-suitable form. These transformations can make use of query-time information to make more informed choices about how to transform objects, and the transformations done for presentation time will be temporal only and not permanently stored. Doing non-permanent reductions at query time makes it less critical that the reductions will not always have high accuracy. At the same time, information about the display of the access device used can be included in the query-time transformations to reduce objects until they are suitable for presentation on that device.

In the rest of this chapter we will first give a system overview, before we specify the requirements for the new version of our framework.

## 3.1 System overview

A digital library system with support for transient digital objects will be developed. The digital library will consist of events and metadata from the SenseCam project at DCU[1], from where we have received a dataset of captured data and metadata by one person in the period from 1st of March to 31st of March 2007. The dataset is further described in Chapter 6. The raw data used is captured using a Microsoft SenseCam device, taking minimum three images every minute for as long as the device is turned on, along with metadata such as time stamp and data from a PIR sensor. In addition we have data from a GPS device used by the SenseCam wearer in the same period, so that the images can be matched with GPS positions by timestamps. In the metadata imported into our digital library, some preprocessing of the raw data has already been performed [7, 8, 9, 10]. In the first of those, images are grouped into events, and it is these events that are imported into our library as digital objects, see figure 3.1.

[1] www.cdvp.dcu.ie/SenseCam

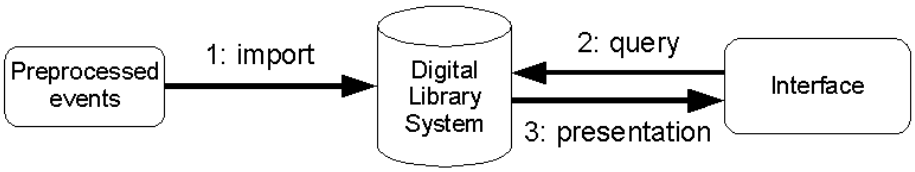



Figure 3.2: Conceptual overview.

The reason for having transient objects is to allow them to be automatically transformed by dynamically configured transformation rules so as to achieve data reduction in the system without losing important information. In a more traditional library approach where digital objects are permanent and static, data reduction can only be performed by object deletion. The transformation rules will get configured during runtime both from system metrics and from the implicit restrictions set by the device users use during the query process. Specifically, events will be analyzed to find duplicate images and images of bad quality.

A web interface is used as the access point to the digital library, and allows users to query for events by their metadata. A ranked list of events will be returned, and by selecting an event from the query result for viewing, the event will be triggered for automatic transformation so as to best fit the device of the user, based on its display capabilities and the query metadata. An overview of the system is shown in figure 3.2.

We are now ready to define the requirements for our framework.

## 3.2 Functional Requirements

**R0: Create objects** The framework shall support loading images and metadata from permanent storage and create digital objects from them.

**R1: Insert objects** The framework shall support inserting newly created objects into the digital library in an unpublished state.

**R2: Read transformation rules** The framework shall support reading in the transformation rules that are configured and add their prerequisites to the monitor logic.

**R3: Monitor digital objects** The framework shall support monitoring the metadata of all objects in the digital library to see if their metadata matches the prerequisites defined in any of the transformation rules.

**R4: Trigger objects for transformation** The shall support functionality for triggering an object for transformation if a positive check was done during monitoring (R2).

**R5: Apply transformation** When an object is triggered for transformation, the framework shall apply the correct transformation rule to that object.

**R6: Provide a multi-modal interface** The framework shall provide a multi-modal interface for accessing and searching for objects.

**R7: Provide a search facility** The framework shall provide a search facility that indexed newly imported objects and make them available through a search interface.

**R8: Present objects in content suitable form** When presenting digital objects to a user after a query, the objects shall be presented in a content suitable form. Unless the digital objects initially suits the client device used, they shall be triggered for transformation until they reach a content suitable form.

**R9: Global data reduction** The framework shall support running data reduction transformtions over the complete dataset based on system metrics such as available free storage space.

## 3.3 Non-functional requirements

**R10: Global reduction accuracy** Only object data of generally low value shall be removed during global reduction transformations.

**R11: Object presentation performance** When objects are transformed on query-time for presentation, the transformed object shall be a better representation of the event than what standard ranking mechanisms can achieve.

# Chapter 4

# Design

This chapter presents the design of our framework. It is based on our previous work, but extended to meet the requirements from the previous chapter. The domain for this work is Lifelog data, and since transformation and presentation of objects are domain and application specific, parts of the design below will reflect this. To make the rest of this chapter more understandable, we will first briefly define digital objects and digital libraries.

Digital Libraries are defined in [16] as a "focused collection of digital objects, including text, video, and audio, along with methods for access and retrieval, and for selection, organization, and maintenance of the collection." Previously being mainly large digital representations of conventional libraries, personal libraries now gain increasing attention due to the proliferation of personal devices capable of capturing digital information during everyday life. In general, most systems that manage both some digital data and their metadata, and provide maintenance and access to these, can be said to be a digital library.

Digital Objects are the key underlying data stored in the digital library. From the concept of physical books and objects cataloged in conventional libraries, digital objects are often wrongly perceived to be fixed and permanent objects. However in the digital world, such objects are malleable, mutable and mobile [17], a property that we will make good use of in our design.

## 4.1 Architecture

### 4.1.1 Decomposition

The Lifelog dataset we have received for use in this thesis comes in the form of images files stored on a hard drive and several database tables with metadata information about the images, both from capture time and from previous works.

A Library Import component will read the specified images and database tables and create new Digital Objects (R0) which will then be inserted into a Digital Library component (R1). A Transformation Engine component will read in Transformation Rules as configured, and send the preconditions for transformation defined in each rule to the Observer (R2). The Observer component can now monitor all the objects in the digital library, and evaluate their metadata against the preconditions defined in the transformation rules (R3).

If a match is found between an object's metadata and the preconditions defined in a transformation rule, a *trigger* command will be sent back to the Transformation Engine with information about the object triggered for transformation, and which transformation rule to apply (R4). The Transformation Engine will now apply the specified rule to the object, and the object will be transformed (R5).

To access objects in the library, a web Interface must be provided. This interface should provide multi-modal access, meaning that the interface should adjust to access from devices with different display capabilities (R6). In addition to browsing through the objects, a search facility should also be provided to retrieve objects from the library (R7). When an object is selected for presentation, it should be automatically transformed until it reach a content-suitable form for presentation (R8).

To maintain advanced tasks, *special transformation rules* will have the ability to reconfigure the preconditions defined in regular transformation rules. This allows for new matches to be found between an object's metadata and the redefined preconditions. By allowing the special transformation rules to be triggered by system metrics, these rules can then reconfigure preconditions such that transformations will be run over the complete dataset. One use of this will be to perform a data reduction on the dataset based on the amount of free storage space available (R9). W call this *global reduction*, and it is a requirement that this reduction process has a high accuracy of object data removal (R10).

When an object is selected for presentation, a query-time transformation will be performed on the object until it reaches a content-suitable form. This transformation process must create a good representation of the object (R11).

### 4.1.2 Overview

The components in the system can now be listed. All components are show in Figure 4.1.

**Library Import (1):** Reads objects from specified input stream or storage medium and send the newly created objects to the Digital Library with object state set to *import*.

**Observer (2):** Monitor object metadata in the Digital Library, and if the metadata matches precondition set in any of the transformation rules, a transform command for that object is sent to the Transformation Engine. In addition, transformation rules themselves can be triggered for transformation, based on object or system state, see *special transformation rules* above.

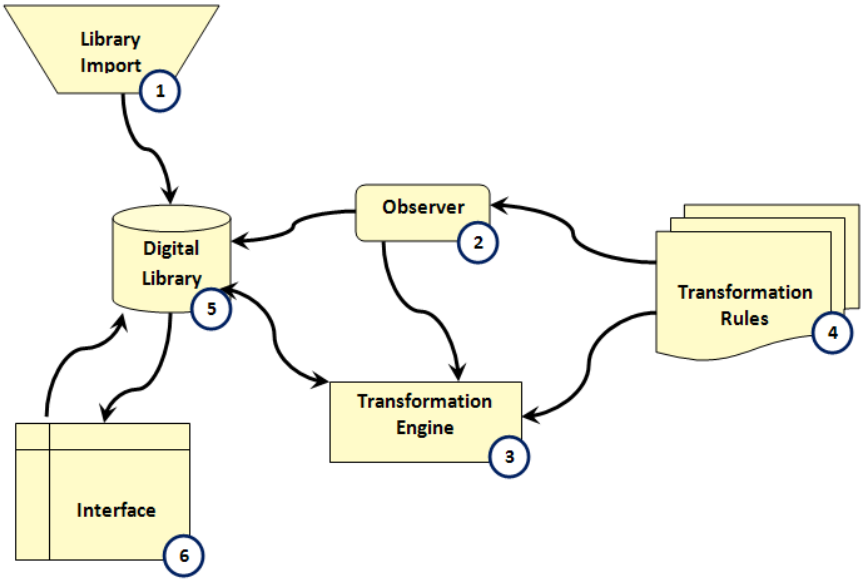



Figure 4.1: Components overview

**Transformation Engine (3):** Performs a transformation on an object, as specified by a transformation rule. When an object is triggered for transformation, which rule to apply will be specified. In addition, rules can also be specified to transform other rules instead of objects.

**Transformation Rules (4):** Consists of an algorithm, a set of preconditions and configuration parameters. Only when the preconditions are made, will the algorithm by applied. The algorithm can apply either to object data, object metadata, or transformation rule preconditions and configuration.

**Digital Library (5):** Provides storage and object access, with respect to the state of the objects - for instance, only *published* objects should be publicly accessible.

**Interface (6):** Provides multi-modal access to the objects in the Digital Library, with both browsing and search functionality.

A detailed explanation of these components follows in the next section.

## 4.2 System components

### 4.2.1 Import system

How objects are inserted into the digital library will be domain and application specific. Objects could be manually inserted by use of a content manager system, fetched from a data stream, or specified for import by a storage media location that for example checks for new files at certain intervals. In general, the tasks of the import system can be described as follows:

1. Read object data and metadata from the specified import method.

2. Create new digital objects in the format supported by the Digital Library.

3. Set the correct object state, and push the objects over to the Digital Library.

### 4.2.2 Observer

The observer's main role is to trigger objects for transformation. To do this, it must first read in the preconditions for transformation defined in each transformation rule. When this is done (normally at system start up), the observer can go on to monitor all the objects in the digital library to see if their metadata match any of the preconditions, and if a positive match is found, that object will be triggered for transformation by the respective transformation rule. This is done by communicating the relevant object and transformation rule to the transformation engine, which should then activate the transformation rule.

The functionality of the observer in combination with the transformation engine is not just used for reducing object data, although that is a main purpose of our system. The observer is also used to manage object state, and handle objects' lifecycle in the digital library. For example, when an object is first created by the import system and sent to the library, the state of the object will be *imported*. Objects in this state will not be available for access. Instead, they will be triggered for transformation by a special transformation rule which purpose is to be sure objects are ready to become publicly available. Such a transformation rule can be set up to do many tasks, some examples are:

- Publish an object on a certain time / date.
- Generate extra metadata for the object.
- Index the objects so they are available for search.
- Filter the objects for example to protect privacy.

### 4.2.3 Transformation Engine

The Transformation Engine receives a transform command from the observer which identifies the objects that are triggered for transformation, and which transformation rule that should be applied. The object is read from the digital library, transformed using the transformation rule, and then written back to the digital library. To make this operation atomic and maintain data integrity, it is performed in a single transaction and the object in the digital library change from published state $P$ to transforming state $T$, so it can not be accessed by other components than the transformation engine during transformation.

### 4.2.4 Transformation Rules

Transformation Rules are the algorithms and procedures that do much of the actual work in our framework, and most of them will be application and domain specific. There are two types of transformation rules: those that transforms digital objects, and those that alters other transformation rules. The first of these was described in our previous paper [3], and contain the following elements:

- Digital object type and transform state identifiers.
- Digital object metadata trigger requirements, which describe what metadata element values are required to trigger the object for transformation by this rule, or system performance parameter values (preconditions).
- The transformation specification which defines exactly how to transform the digital object from it's current state into another.

The transformation specifications can in some cases be as simple as a 'delete' command based on the object age. However they can also be advanced algorithms that analyze the object data and transform it into a reduced or otherwise altered representation.

This basic model of the transformation rules was sufficient for our previous work, where the prototype only dealt with simple CCTV data captured from a static environment. With more advanced data, as in the Lifelog context of this thesis, a more dynamic system became necessary, where the transformation rules themselves can be changed by certain conditions. We therefore introduce a new type of transformation rules, which do not alter digital objects, but instead alter the configuration of other transformation rules. These transformation rules contain the following elements:

- Transformation rule type and configuration values.
- Transformation rule trigger requirements, which will can be either digital object metadata values or system performance parameter values.
- Transformation rule reconfiguration specification, which defines exactly how to alter a specific rule.

An example of a specific use-case for these new transformation rules is the global reduction that are performed in our digital library when the available free storage space in our system gets too low. A transformation rule will then modify the configuration of the quality and the similarity transformation rules by increasing their thresholds for activation. This again will have the effect that new objects are triggered for transformation by the observer, and a set of objects will be reduced by the quality and similarity transformation rules. Until the requirements for free storage space are met, the thresholds of the two transformation rules will continue to be increased; this therefore creates the *global reduction loop* as described in the beginning of this chapter.

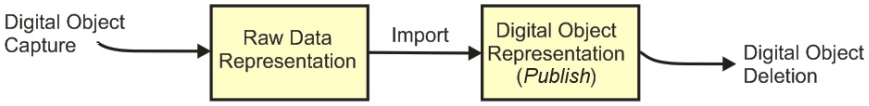



Figure 4.2: Conventional Digital Object lifecycle, from [3], Figure 1.

### 4.2.5 Digital Library

The Digital Library component is the part of the system that provides storage for and access to digital objects. A typical storage solution will be a combination of a storage medium for object data, and a database for object metadata. Access will be available both by manually selecting objects and through a search facility. Access requests will be made from the interface component, and an important feature of this framework will be the support for interfaces of different modality.

More traditional digital/electronic library systems tend to focus on detailed access facilities tailored for a computer workstation. However, today's proliferation of multimedia-capturing devices into personal and everyday activities, a modern digital library may contain data suited for a range of different situations and contexts, where the device used for access may not have the capabilities of a computer workstation (or may even surpass it). Access in our framework will therefore be able to receive or implicitly register the context and the capabilities of the device used for access, and to use this information to deliver objects to the device's interface in a content-suitable form.

The framework presented in our previous paper extended the standard digital library lifecycle model shown in Figure 4.2 to support transitive digital objects so that we could perform data reduction on the objects instead of deleting them (Figure 4.3).

For the digital library to be able to make use of query-time information such as context and access device capabilities, the model is extended with a new evaluation loop where objects can be triggered for query-time transformations until they reach a content-suitable form, shown in Figure 4.4. When a content-suitable form is reached, objects can be presented in the interface.

One example where this is put to good use is if you are accessing a Lifelog digital library using your cell phone. Lifelog data is best understood as events, which will be imported as objects in the digital library, and object data will mostly consist of images captured in a sequential matter. A Lifelog system is used as a Human Digital Memory, and a view of the objects must therefore

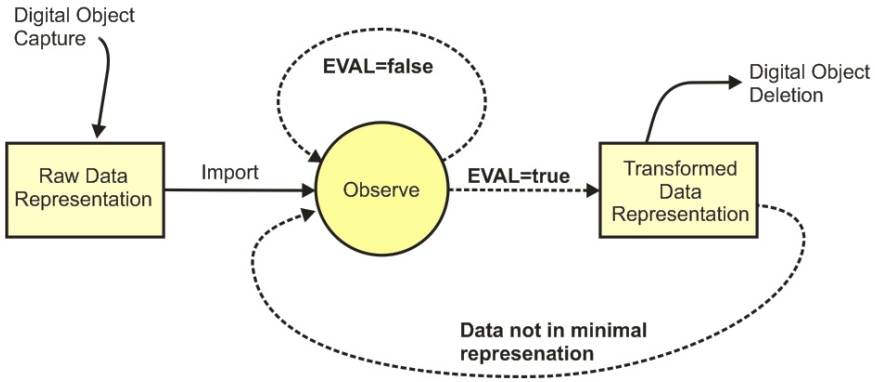



Figure 4.3: Digital Object lifecycle extended to support transitive objects, from [3], Figure 2.

present the event as a whole as good as possible. Small-screen access devices like a mobile phone will not be able to view all of the images of an event at once however, and navigation and feedback is often cumbersome to perform. In our extended framework, the digital library will receive information about the display capabilities of the device used for access, and the object can be transformed at query-time until it reach a form suitable for display on a small-screen device; for example can a set of only four images be presented, instead of up to 500 images as in some events. When a search is performed, all objects in the search results will be automatically triggered for query-time transformation. When such an object is selected by the user for presentation, a presentation time transformation rule will use query-time information such as access device modality and search terms to configure transformation rules that will transform the object into a content-suitable form before sent to the user for presentation.

In addition to improving the presentation of an object at query-time, a reduction of object data sent over the network is also achieved. To improve the user experience, a system can be configured to use the user context as search terms. For instance, a query will automatically be constructed using time of day and geographical location of the user as search terms, which will be possible with many modern mobile phones.

### 4.2.6 Interface

The interface component provides access to the objects in the digital library in two ways: manual look-up of a specified object, and through a search form where relevant search terms can be specified. Additionally, the interface must be able to provide information about the modality of the device used for access, such as mobile phone, TV, workstation computer etc. A natural choice for a network-capable interface today is a web-based interface, which can adapt to different screen sizes.

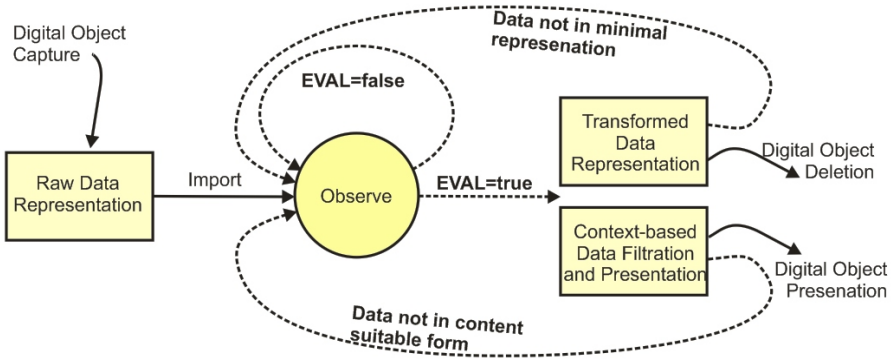



Figure 4.4: New proposed extension to support query-time transformation to a content-suitable form

## 4.3 Summary

A design of a system has been presented that will fulfill the requirements specified in Chapter 3. Digital Objects will be read created by the Import system and sent to the Digital Library in an *import* state. Throughout the object lifecycle it will be monitored by the Observer, which will evaluate the object metadata and system performance in comparison with object state identifiers and parameter values as specified in the transformation rules used in the system. Upon a positive match of an object and a transformation rule, a transform command will be sent to the Transformation Engine, and the object will now be triggered for transformation. The Transformation Engine will in an atomic operation to maintain data integrity read the object from the digital library, apply the operation or algorithm specified by the transformation rule, and write the transformed object back to the digital library. The original framework presented in [3] have also been extended with a new special transformation rule as described in 4.2. An important extension has also been made to the object lifecycle model such that a special set of transformations can be applied to an object at query-time using information dynamic to the context and query-time parameters to transform objects until they reach a content-suitable form before presentation. A web-based user interface supports multi-modal access.

# Chapter 5

# Prototype Implementation

## 5.1 Implementation environment

The implementation environment consists of a Dell Precision workstation running Windows XP-64. C# ("c-sharp") is used as programming language for the prototype implementation, and the web interface is created using .NET ASP 3.5 running on a IIS 5.1 server on localhost. The image data is stored on a local hard drive, and object metadata is stored in a local MySQL database.

## 5.2 Interesting Details

As a prototype implementation developed for the purpose of being able to run the experiments specified in the Chapter 6, most of the code is practical rather than beautiful. A few details however may be worth mentioning.

### 5.2.1 Run-time cache

Although the dataset used for this thesis can not be considered to be very large, it does contain a lot of metadata, spread out on multiple tables and databases. Since metadata is used in almost any operation performed, the adoption of caching throughout the logic is extensive. Most of the cached data regards the digital objects during their lifetime in the library, and the library component therefore have a static Cache class with C# Dictionary and List objects used as memory storage. The other main usage of cache is in the Search class, which contains temporary cached objects from the search results.

This provides for fast and comfortable access to cached data using LINQ[1]. In the example below, non-removed images for an event is sorted by rank after a presentation time reduction has been performed:

```
var images = from image in Event.images
             where image.State != "removed"
             orderby Query.Rank[image.ID] ascending
             select image;
```

The images variable will now be a *List<EventMember>* object. EventMember is a special class used to contain presentation-time information about each event member (each image is together with it's metadata called a *member* of the event in this implementation).

### 5.2.2 Search engine

Although not evaluated in this thesis, a detailed search form has been developed and is provided in the interface. As the underlying search engine, the open source Lucene.NET search engine library[2], a port of [3], has been used. Lucene is a high-performance full-featured text search engine.

When Digital Objects are imported, their metadata is indexed in Lucene by a total of 8 fields, namely

*Geographical Location*
*Geographical City*
*Geographical County*
*Geographical Country*
*Context Descriptors*
*Time of Day*
*Combined Field*

All the geographical information is collected by a geo transformation rule originally applied on each object during import. First, timestamp of each image in each event was matched to the gps data collected by the GPS receiver. If no matching timestamp existed, the closest value was selected. The gps coordinates could now be looked up in a local copy of the GeoNames database[4] to find the geographical information. The collected geoinformation was later added to the permanent storage for each object, and the transformation rule was removed from the configuration.

The search engine interface is shown in the Interface section below.

[1] http://msdn.microsoft.com/en-us/vcsharp/aa904594.aspx
[2] http://incubator.apache.org/lucene.net
[3] http://lucene.apache.org/java/docs
[4] http://www.geonames.org

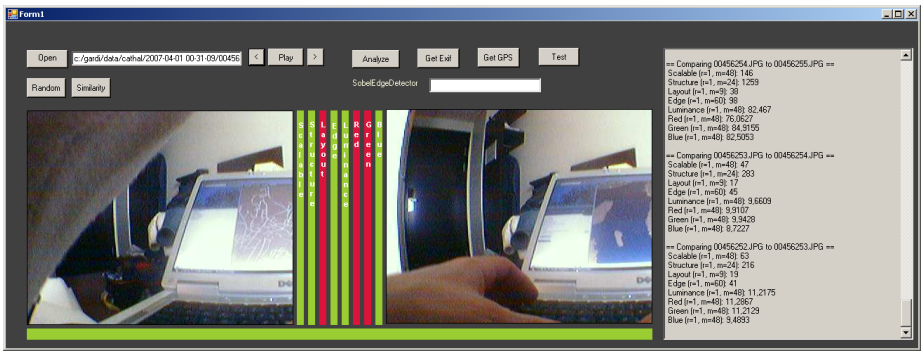



Figure 5.1: Image Similarity debug interface.

## 5.3 Debugging

During development of the similarity algorithm which decides whether two images are visually similar or not, a Microsoft Form project was created to help visualizing the performance of the current implementation. Various other debug tasks could also be performed from the interface.

Figure 5.1 shows one of the later versions of the debug interface. At this point of development, eight different image metrics were analyzed and compared between the two images, visualized by the vertical bars between the images. Each image metric have different threshold for similarity, and the corresponding bar is shown as green if the metric delta between the two images were below the threshold, and red if the delta was above the metric. Below the two images are a long horizontal bar. This shows the overall score of the similarity test. At this time of development, the images would be measured as similar of four of the individual metric deltas was below the threshold. To the right in the figure we can see a scrolling text log from the image analysis.

## 5.4 Prototype Interface

The interface was developed as a separate .NET ASP project which included the framework implementation as a DLL. Standard HTML code was used to structure the web documents, and layout was created using the Blueprint-CSS framework[5]. Except for the search interface, all other views take an object as input through a HttpRequest parameter called *id*. A typical lookup for an event would for example be *http:/localhost/GardiWeb/Event.aspx?id=6189*, which would display the event (digital object) with the internal ID of 6189.

Figure 5.2 shows the search interface with query results in ranked order. In this example, a free-text search has been performed with the inclusive search terms *+wicklow*, *+outdoor* and *+veg*. *Wicklow* is the geographical name of an area in Ireland, while *outdoor* and *veg* (vegetation) are concepts detected in each image, from the work described in [8]. A plus sign has been included in front

[5] www.blueprintcss.org

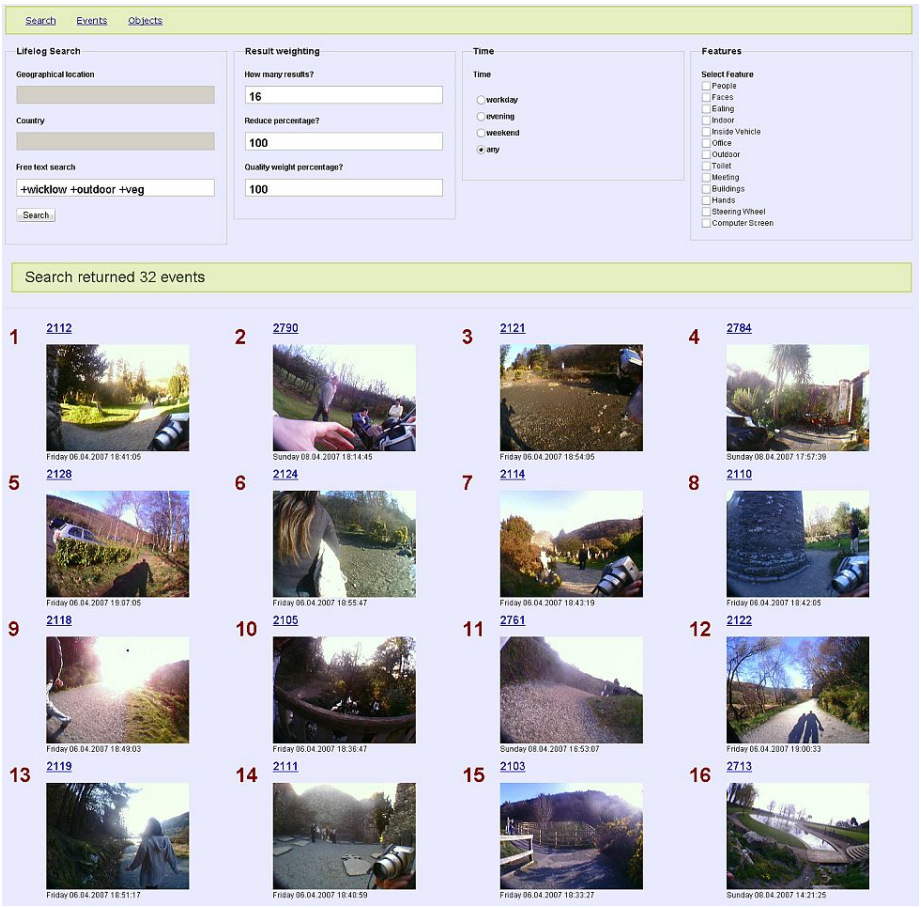



Figure 5.2: Search interface with display of search results in ranked order after a query has been performed.


Figure 5.3: Evaluation interface with removed images.

of each search term, which means that only results indexed by all three terms will be returned. In the "Time" input field, *any* is selected to allow for objects captured at any time.

In Figure 5.3, an evaluation view of an object is shown. We see an object viewed after a search has been performed. Modality has been specified to that of a mobile device only capable of displaying four images. In addition, a global reduction of 40% has been performed on the complete dataset. In the view we see the images that are part of this event, sorted by time, and each image have a colored border. Images with a red border was removed during global reduction, while images with a yellow border was removed during query-time transformation. Only images with a green border would thus be displayed in a normal event view. We see from the numbers above the images that the event originally had 76 images, now reduced to 4 after the presentation-time reduction. On top left we see a slightly larger image which is the highest ranked image of this event, and to the left are some additional metadata. This experimental view was used under the accuracy evaluation.

# Chapter 6

# Experiments and Evaluation

This chapter gives a description of the experiments conducted for this thesis. First, the right measure for our Lifelog filtering system is identified, before the results of our experiments are presented.

## 6.1 Identification of Goals

This thesis makes the conjecture that automatic filtering techniques for data reduction can be applied to advanced data sets such as the captured data from a Lifelog system, thus increasing data quality and reducing data redundancy without human input. This will improve the usability of such multimedia archives.

To identify the correct measurements for our system we fill first identify the goals of our system, then precede to define the success for each goal, before we present the measurements that are performed to evaluate our goals.

Two of the goals for this system are tightly connected: Reducing storage requirements, and increasing performance of queries. Storage requirements are reduced by transforming objects to reduce their data size by finding and removing duplicate images or images of bad quality. This should then also increase the performance of queries, since the objects we query over will contain less data in need of processing.

The data set consisting of the captured image data from a Lifelog system is advanced both in context and semantics, and it is personal to the wearer of the SenseCam capturing device. Because of this, it is hard to make general filtering rules that can safely remove redundant or bad quality data with high accuracy. Instead, most of the data reduction is done at presentation time of query results. This enables us to display only as much data as the user can view on his/her presentation device, while also reducing the amount of data transferred to this device.

For the global reduction, which is used whenever the system requires more free space, there are no specific interests that can be taken into consideration while deciding how to reduce the objects. Data reduction at the global level should therefore focus on having a high general accuracy and only remove data of clearly bad quality or of high similarity to other data.

Data reduction at presentation time will be forced to achieve a specified amount of reduction, but will have query-time information available to help select the best data for reduction. Our conjecture is that under the constraints of presentation devices of limited display capabilities, doing multiple data reduction loops with focus on data quality and similarity will enable us to give a better view of each event than a standard ranking search engine can do.

The identified goals can now be summed up:

1. Achieve data reduction both globally and at presentation time.
2. Data reduction should also give increased query performance.
3. Global data reduction should have a very high accuracy.
4. Presentation time data reduction should perform better than a standard search engine could.

For the goals of data reduction, *how* it is achieved makes all the difference - simple approaches would be to delete the oldest data, or random data. Here, for goal 1 of data reduction to be successful, it must also meet the criteria of goal 2, 3 and 4. Nevertheless, the system must also be able to actually achieve the desired data reduction.

For the goal of increased query performance, success can be evaluated by comparing the decrease of resources used by a query when running it over a dataset that has undergone different levels of data reduction.

Success of global data reduction accuracy must be evaluated by inspection. One can expect that global data reduction can be done with high accuracy up to a certain amount, after which it will start to decrease, and that this will be user and dataset-specific: A different dataset with a different distribution of data quality and data similarity will allow for other accuracy performance.

Likewise for the presentation-time reduction, this will be dataset-specific, but also highly specific to the quality of the metadata available, as this is used in the decision process during data reduction at this time. Further, some events will contain data that is easily reduced, while others would require much more advanced image and metadata analysis before data reduction can be done at high accuracy. Success will be measured by evaluating the presentation-time reduced view of an event as created by this system, compared to a pure ranking-based view based on what traditional search engine can do.

## 6.2 Dataset and Metrics

The dataset used is from the Lifelog project at DCU, where Cathal Gurrin during the last 20 months have gathered a HDM archive of almost 3 million photos, including the following data:

- Photos (4,500 per day)
- GPS locations to the nearest 5 meters approximately every ten seconds he is moving outside DCU
- Temperature
- Accelerometer motion Light/Brightness

This dataset is very rare and one of the few available at this size. Not many persons have yet subjected themselves to wearing a camera constantly for such a long period of time. In this thesis data from the period of 1st of March 2007 to 31st of March 2007 is used, consisting of 79 595 images. Additionally, the following metadata from other works have been added to our system:

- Event segmentation, where a total of 1071 events were created from the 79 595 images [7]. These events are used as the objects in our system.
- MPEG-7 descriptor values for the following: Colour Layout, Colour Structure, Scalable Colour and Edge Histogram [10].
- Semantic concept detection for all images [8].
- Image Quality measures [9].

In various experiments, different degrees of reduced data-sets are used, where bad quality data has been automatically filtered out. Additionally, a temporary subset is created for each query. Some of the experiments were run on a limited set, where during data import to the library, only 15 000 images were allowed to load. These images spans over the 271 first events, out of 1071 in total.

Each time a filtering loop is run, either over the complete set of data or over a query subset, the set in question is reduced so it contains a subset of the previous data. For temporary data-sets created on query-time, these subsets created are dependent on query parameters, while for reduction loops on the complete dataset are run based on (simulated) system metrics such as remaining, free storage medium space.

Only the person wearing the camera can be the inspection judge in the evaluation process.

To better understand some of the results, a rough profiling of the dataset was done, where the distribution of image data per event and the distribution of image quality was measured.

Figure 6.1 shows the distribution of image data per event. We see that most events have somewhere between 1 and 100 images. The average amount of images per event is 83, however this is a bit skewed by four events with more than 575 images in them.

Figure 6.2 shows the distribution of the image quality measure. Most images falls into the category 0.0020 - 0.0039, even though the theoretical values could be as high as 1.0000, we see that most images have a measured quality value below 0.0319.


Figure 6.1: Distribution of image data per event

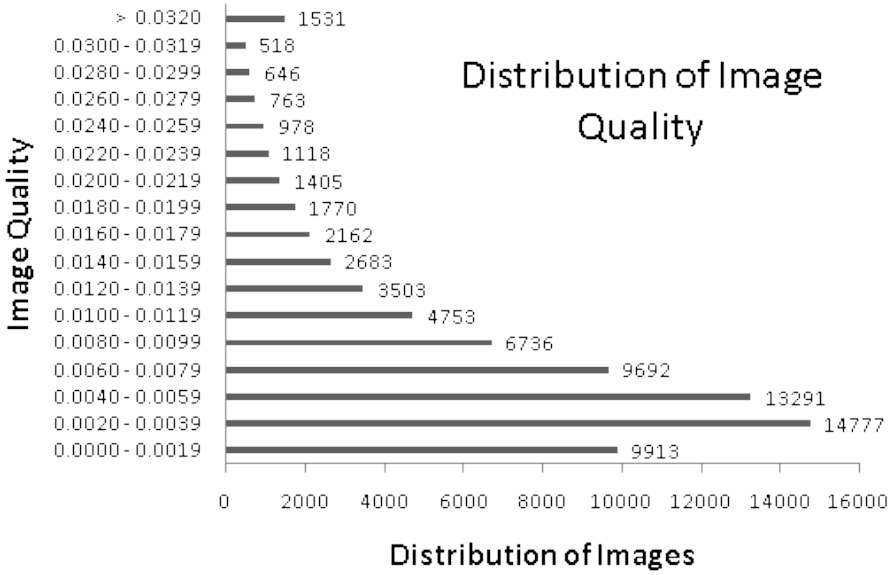



Figure 6.2: Distribution of image quality

The metric for measures for storage space used by the different data-sets is read using .NET's *FileInfo* class and saved to logs during runtime, timing is done using .NET's *Stopwatch* class while accuracy is evaluated by manual inspection during experimentation by the wearer of the SenseCam.

## 6.3 Experiments & Results

The experiments were performed on a Dell Precision 390 Workstation with Intel Core2 Quad Q6600 CPU, each core running at 2.40 GHz. The workstation was equipped with 4Gb DDR2-6400 RAM and a Western Digital 500Gb hard disk with 16Mb cache and 7200 RPM.

Microsoft Internet Information Services 5.1 was used to run the web application developed, with ASP .NET State Server service running. Images were stored on the hard disk, while all permanent metadata was stored in and accessed from a MySQL 5.0.51b server running on localhost.

Images for the Lifelog project was captured with a Microsoft SenseCam camera which captures images in 640 x 480 pixels resolution and stores them using JPEG compression.

### 6.3.1 Data reduction

This experiment is connected to the general goal (1) of reducing data storage requirements when necessary. An example scenario would be a digital library system with limited resources, where disk usage and incoming data flow is monitored. When free disk space becomes too low, or alternatively at scheduled times, the complete dataset can be triggered for transformation. Each object will then be sent to the transformation engine, where it will be analyzed and maybe transformed, according to the transformation rules. For such a scenario, the transformation rules will dynamically change during runtime so that they can free up as much space as necessary. In our domain of life logging, one of the transformation rules inspects the images of an event to figure out which of the images have the lowest image quality. The transformation rule contains a threshold for image quality, and will reduce the data of the object (event) by removing image data for the captures that fall below the set threshold. However, next time the system needs more free disk space, a new threshold is needed for the transformation rule for it to have any effect. This threshold is therefore set dynamically during runtime.

The experiment simulates the effect of a system requesting more disk space by running the quality transformation rule and the redundancy transformation rule over the complete dataset in a loop where the respective thresholds are dynamically set until the requested amount of free disk space is acquired. The system will be forced to reduce the dataset in 5% increments from 5 to 40% percent reduction, to evaluate how the system responds to a simulated requirement of more free storage space. Some metrics about the reduction process will be measured and presented, and the experiment is performed on the complete dataset

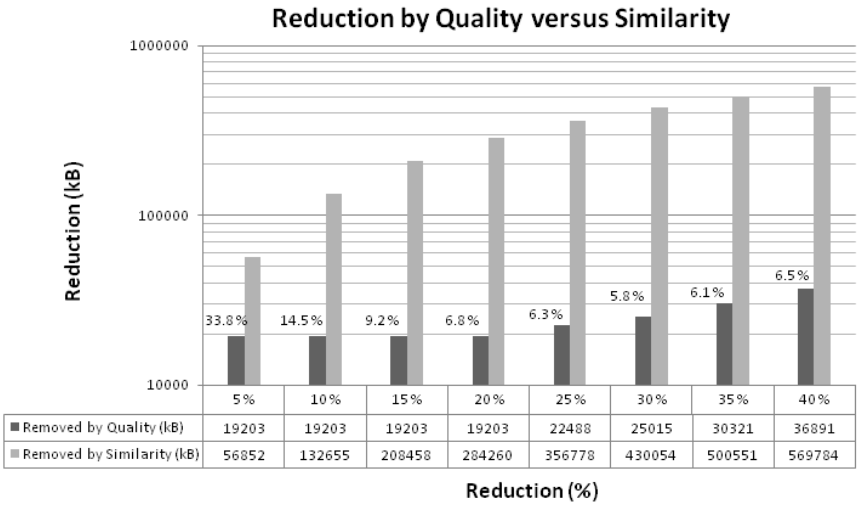



Figure 6.3: Data reduction by quality versus similarity

of 1071 objects containing 79 595 images. Accuracy evaluation for the same setup is discussed below.

During global data reduction, where the system is forced to reduce the total data size with a certain percentage, reduction amount is tied to the size of the objects. For the Lifelog dataset used in these experiments, most of the object data consist of captured images from the SenseCam. It was observed that each object contained on average 83 images, and further that the average image size was 19.9 KB. Reducing the dataset by one percent will thus free up around 19 Mb of storage space.

Figure 6.3 compares reduction by the two transformation rules Quality and Similarity, where the reduction in KB is showed on a logarithmic y-axis and the reduction percentage is shown on the x-axis. Above each Quality column is shown the fraction of reduction by quality versus similarity. We see that from 5% to 20%, reduction by quality stays the same, which can be explained by the thresholds chosen for the two transformation rules in this experiment: The thresholds for image similarity does not have to be increased until the system asks for a 25 percent reduction. Thus, from 5 to 20 percent, only one reduction loop is needed, and the quality reduction is performed only once using the starting threshold for that rules, which does not give the wanted reduction in any of the cases such that the similarity transformation is performed after wards. This is also seen by the quality reduction being a bigger fraction of the overall reduction at the lower percent than at the higher percent.

Figure 6.4 shows the number of global reduction loops required per reduction percent. Reduction were done in increments of five percent. We can see that it gets harder and harder for the system to achieve the data reduction required, forcing a reconfiguration of the transformation rules to increase their thresholds, and a subsequent re-run of the transformations. From 25% reduction, the amount of re-runs needed seems to increase linearly.

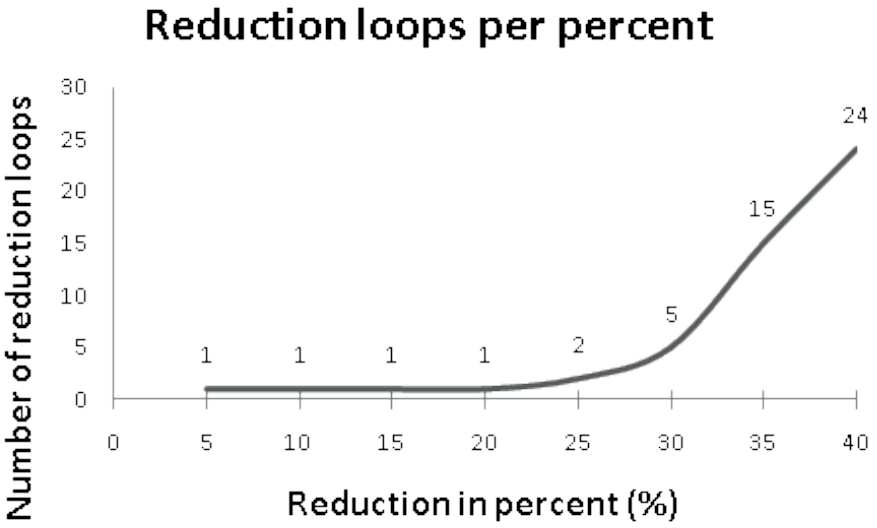



Figure 6.4: Reduction loops per reduction percentage

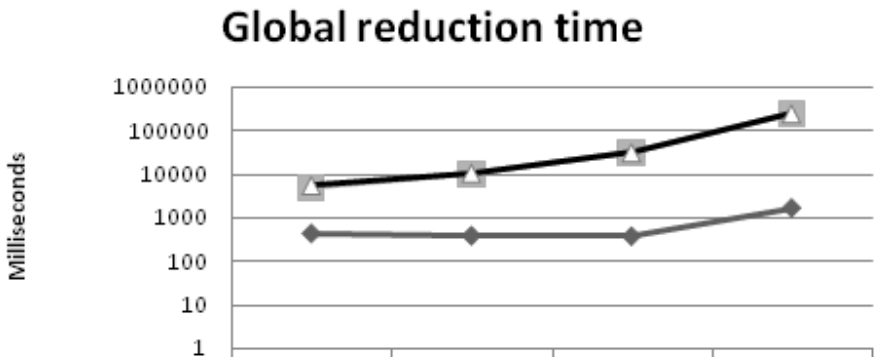
### 6.3.2 Performance

When we transform objects so their contents contain less data, one of the benefits should be that subsequent queries and transformations require less system resources. Since all experiments are performed on the same computer with the same environment, we will measure the response times of a set of queries after different levels of global reduction was performed. This should then loosely be tied to reduced use of system resources, mainly CPU utilization. CPU utilization, use of random access memory or use of disk cache will not be measured specifically. This experiment is performed on a reduced dataset containing the 271 first objects (out of 1071).

After each subsequent global reduction, a specific query will be performed and the same set of objects will be selected for view in the user interface. This trigger the objects for presentation time reduction until the objects reach a content-suitable form based on the capabilities of the access device used. The time spent in each transformation rule as well as the total time of the presentation-time reduction will be measured, along with some other metrics. The access devices simulated will be a mobile device, an eeePC[1] netbook, a HD TV and a computer desktop monitor, where the numbers of images each device is capable of viewing on one screen is set to 4, 9, 20 and 50 respectively.

The transformation rules used in the system will be both domain specific and application specific, and a transformation can have many purposes. Excluding the terminal transformation where an object is triggered for deletion so that its life-cycle state is set to *deleted*, most transformation rules however will either generate or modify metadata, or reduce object content data size. It is the latter that should have their performance the most affected by data reductions. In the

[1] eeePC.asus.com




| | 5% | 10% | 20% | 40% |
|---|---|---|---|---|
| Quality (ms) | 430 | 380 | 370 | 1610 |
| Similarity (ms) | 4870 | 9900 | 30240 | 244250 |
| Total (ms) | 5300 | 10280 | 30610 | 245860 |

Figure 6.5: Time per global data reduction in percent for the quality and similarity transformation rule. Note the logarithmic time-scale.

prototype implementation created for this work there are two such transformation rules: the quality transformation rule and the redundancy transformation rule. The quality transformation can be expected to change linearly with a reduced data set since it compares each image in an object independently, while the redundancy transformation should be expected to change non-linearly since it compares each image to the other images of the object's data content.

Figure 6.5 shows that the time spent on global reduction increases logarithmically, and that the majority of the increased time is spent doing similarity reduction. We also see that the time spent doing quality reduction is almost the same for 5 to 20 %, which is explained by Figure 6.6, where we see that the same amount of images is removed by quality for these percentages. This is the same effect that was shown in Figure 6.3.

The huge increase in time is also strongly linked to the thresholds configured - less strict threshold have been shown to perform linearly, but then with much reduced accuracy. Combined with the results shown in Figure 6.4 above we understand that for the smaller reductions, a first loop does not need to run over all the data to achieve the reduction goal, and that first at 25% reduction requirement will another reduction loop be run. It should also noted that no focus has been given to load balancing or multi-threading in the prototype implementation.

### 6.3.3 Accuracy

These experiments were performed on the limited dataset of 271 events.

In this experiment we will evaluate the quality of the selection of data for removal during the global reduction. During global reduction both quality trans-

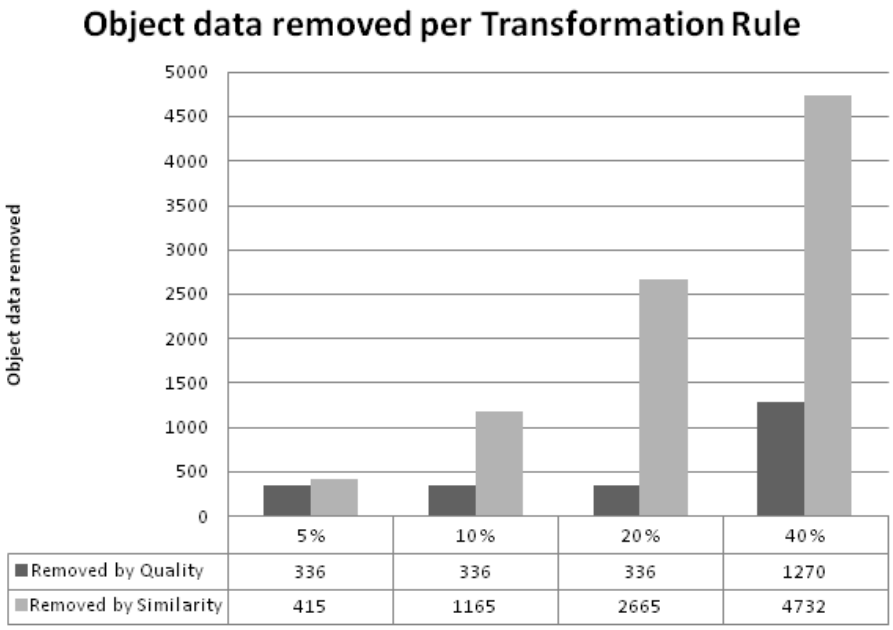



Figure 6.6: Object data removed per Transformation Rule.

formation and duplication transformation is applied, and we will measure the combined effect of the two transformations over the whole object. A random selection of 25 objects are made, and for each of these objects we will evaluate the accuracy of data removal of a 5, 10, 20 and 40 percent reduction respectively.

The question answered during evaluation is for each image in the object: Was this image useful or was it OK to remove it? The evaluation will be a human judgment of accuracy made by the owner of the dataset.

This experiment evaluates the object transformation done at query-time where the objects are reduced until they reach a content-suitable form. Search terms from the query are used in the transformation of the object data, along with the other available metadata about the object. The object is said to reach a content-suitable form when the object data has been transformed to fit the devices used for library access.

A random selection of 10 objects out of the 25 used in the last experiment will be made for this experiment. Since a query is necessary to produce the query-time information, a query is selected for each of the 10 objects such that the object will get a high ranking in the search results. A global reduction of 40% have been performed up front of this experiment.

The question answered in this experiment is how our systems presentation of an object compares to a baseline view and an advanced search ranking view. The baseline view compared against is a time-ordered view of the images in the event. The advanced ranking is similar to what a modern search engine could achieve, where each image in the event is given a rank based on image quality and detected situations, where situations that were part of the query receive a

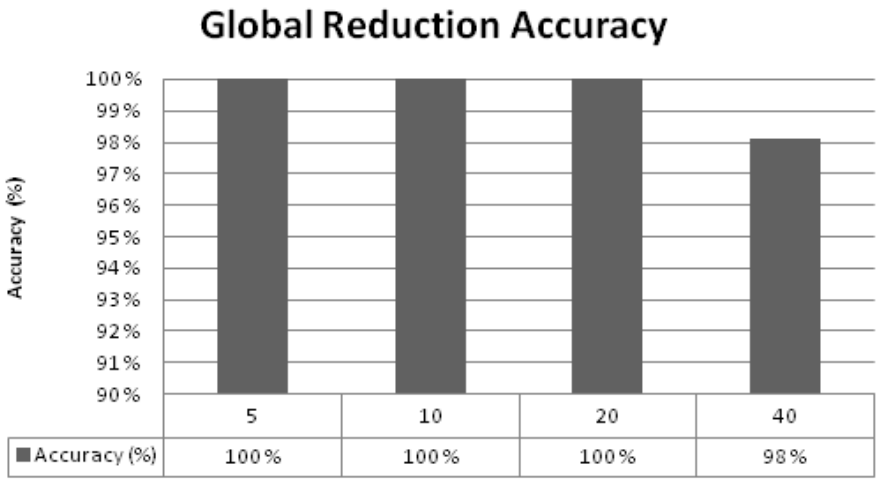



Figure 6.7: Global Reduction accuracy evaluation at 5, 10, 20 and 40 % global reduction.

higher ranking. Our system use the same ranking, but additionally does data reduction to remove both redundancy and low-quality images.

For each of the views (4, 9, 20 and 50 images), a rating is given as to how well the view represent the event, on a scale from terrible through bad, average, good and to excellent. The rating is personal to the owner of the images.

Figure 6.7 show the evaluation of the global reduction. In each of the reductions of 5, 10, 20 and 40 %, a set of 25 objects were reviewed by the owner of the images. A good / bad rating was given for each removed image of an object, with good meaning that the image removed contained no valuable information for the event, while bad means that the owner think the image should not have been removed. If for example seven images were removed for a given object at 40% global reduction, and the owner rate one of the removals as bad, the accuracy for that event becomes 6/7 or 0.86 %. The average accuracy over the 25 selected objects is then taken as the accuracy of the global reduction at the specified reduction percentage.

In the figure we see that a very high accuracy was achieved in this experiment. For a reduction of all the data in the Lifelog dataset of 5 to 20 percent, 100% accuracy was achieved, which means that no valuable information was lost during this data reduction. At 40% global reduction, an accuracy of 98.09 % was achieved.

Figure 6.8 shows the average performance for the presentation-time reduction done for four different modalities, namely Computer screen, HDTV, eeePC and Mobile device. Our system (Gardi) is compared to that of a traditional search engine, and a base-line time-ordered view. Each presentation of the object was rated as either terrible, bad, average, good or excellent. These ratings has then been converted to a scale from 1 to 5, and the average rating for each view has been calculated.

We see that the rating generally declines as the presentation-time reduction is

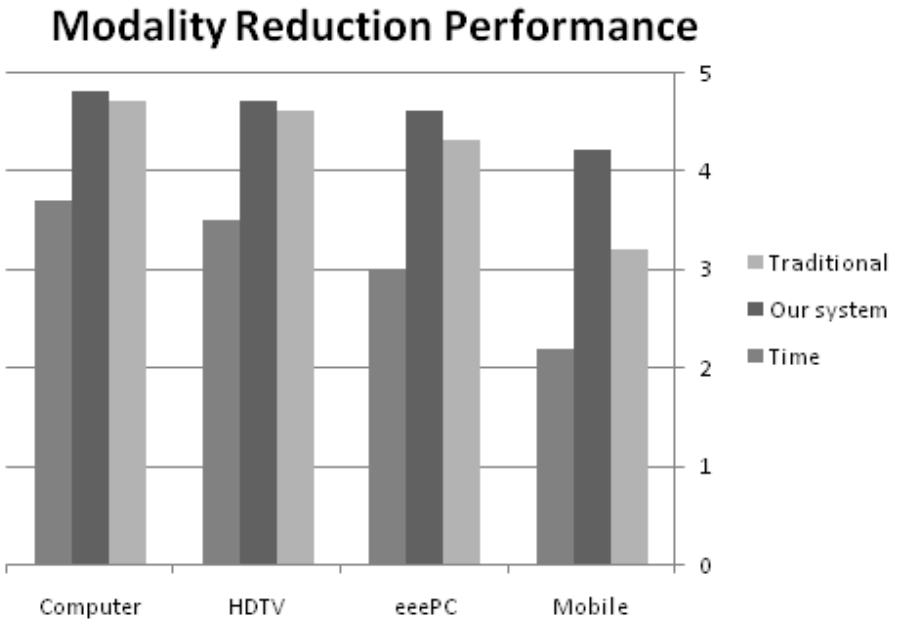



Figure 6.8: Performance of presentation-time reduction for different modalities.

increased to support smaller-screen devices, and that the time-based baseline view almost always is considered the worst presentation of the object. Compared to a traditional ranking of the object data (images), Gardi performs increasingly better as the demands for reduction gets higher, and Gardi always outperforms the other two presentation models. It is noted that for a computer screen which we defined could have 50 images in one view, the rating is generally high. One reason for this is that for many of the randomly selected objects in this experiment, the amount of image data was less than 50, such that no reduction took place and only the general ranking had effect. Some of the underlying data also shows that the time-based view performs best with very few images in the object, while Gardi and traditional ranking performs equal when no little or no reduction takes place.

## 6.4 Evaluation

From the requirements in Chapter 3 we see that the system must be able to automatically transform objects to the most content-suitable form during presentation (R8). The query-time transformations of R8 are not only about how to best represent the objects on different screen sizes. One important part of it is also the **data reduction** performed when an object for example is reduced from containing 78 images to one containing only 4 images, as seen in the experiment screenshot in figure 5.3. For that example, we can estimate the data reduction to be $74 * 19.9\,kB \backsimeq 1.5\,MB$. During this reduction though we are not asking the system to give us the best reduction it can do while retaining

| | M(50) | M(20) | M(9) | M(4) |
|---|---|---|---|---|
| $I - M$ | 33 | 63 | 74 | 79 |
| Data Reduction | 0.66 MB | 1.3 MB | 1.5 MB | 1.6 MB |

Table 6.1: Average presentation time data reduction for event per access device modality when $I$ is set to be 83.

good accuracy, but instead force to achieve the following reduction: Let $I$ be the number of images in the event, and $M$ be the number of images that can be viewed on a device of a certain modality. The required data reduction will then always be $I - M$. We can now estimate the average amount of data reduction performed for each of the four modalities used in these experiments, shown in Table 6.1.

The requirements also specify that it must be possible to achieve global data reduction over the dataset if required by the system, for example when low on free space (R9). Originally this was believed to be small reductions between 0.1 and 1 % performed on a large dataset. While doing the accuracy experiments though it became clear that a much larger reduction could be performed globally while still maintaining very high accuracy in the data reductions. We therefore used reductions from 5 to 40 percent and achieved good results. It is clear from Figure 6.3 that in the current implementation and transformation filter configuration, by far the most reduction is performed by the similarity transformation. Coming from our simple video surveillance environment described in our experiments in [3], the more advanced data of lifelogging was believed to be much harder to reduce by removing redundancy. Surprisingly, evaluation of the global reduction tests in Figure 6.7 shows that even at 40 % reduction, accuracy is as high as 98.09. We have already concluded above that the majority of the reductions are performed by the similarity transformation rule. We can therefore acknowledge that the image similarity filter developed in this work performs very well, and that the requirement of global reduction accuracy has been met (R10). The recall of the image similarity filter have not been evaluated, but we believe from the results shown above that the algorithm have the potential of removing redundant images with 100% recall. How to automatically find the certain set of filter thresholds required to reach 100% recall for each event, however, is not known, and may not be feasible to do. We also see from Figure 6.5 that although a 40% data reduction could be achieved with high accuracy, an increasingly large performance penalty is introduced as the reduction requirements increases. An interesting experiment in later work will be to measure the relation between accuracy and performance during reduction, with a set of different threshold configurations for the transformation rules.

In Table 6.1 we showed the reduction performed during query-time reductions. The requirements also specify that this reduction should transform the object into a better representation than what can be achieved with standard ranking methods (R11). Evaluating this is very hard, since there are no other systems where this dataset can be evaluated so a direct comparison to other systems can not be performed. Further, it is not necessarily known what the best representation of an object is. It will change with the access device used, and we try to evaluate this by simulating four different access devices. However, the

representation of the event can only be evaluated as perceived by the owner of the data, and no work currently show if this perception changes over time, with mood, or with context. We will have to assume that the perception did not change during the experimentation phase. In our experiment, we chose to use a time-ordered view as the baseline representation, known for example from Google Picasa photo album software[2]. This baseline could then compare to a standard ranking view and our query-time reduced view. Although anecdotal with only 10 events evaluated, several interesting aspects seems to emerge. For all modalities (mobile device, eeePC, HDTV, computer monitor), our system makes better representations than a standard search ranking, and the time-ordered baseline gives the worst representation. We can also see that all representations performs worse as the display capabilities of the access device decrease. Our system however seem to handle the increasing restraints introduced by smaller-screen devices better than the other presentation models, and the presentation by our system is on average always rated above "good." Although no facts can be drawn from this small-scale experiment, the tendency shown is that our system fulfill the requirements of R11.

### 6.4.1 Effect of configuration

While the results seems to show a well-performing system, one should have in mind that most of the results are highly dependent on the configuration used during experimentation, and most notably that of the transformation rules: the selected thresholds for image filtering and the steps by which the thresholds are increased have very measurable effects on the performance. For example, by increasing the thresholds for the similarity filter, in many cases only one reduction loop will be necessary to achieve the desired data reduction, however with the risk of losing accuracy. One of the reasons behind this is that the check for similarity can not be randomized: this image metric is specifically tailored to match images that are nearby in capture time. This filter therefore start with the first capture image in the event and compares it to the neighbors, then move on the the next image, and so on. By using loose thresholds, the desired reduction would have been achieved by removing images only from the start of the event, while there may be images later in the event that have a higher degree of similarity that would not be removed.

Many events also contains a lot of similar images that are easily filtered by the similarity transformation rule. On global reduction loops, the desired global reduction could then be achieved by reducing only a limited set of events, instead of evenly reduce all of the events. An earlier attempt at stopping this from happening was to limit the possible amount of reduction in each event for one loop (instead of lowering the thresholds), however this then opened for the effect described above where only the beginning of the events where reduced. By reasoning, the most even results should be achieved by using very strict starting thresholds, and by increasing the thresholds in very small steps. Preliminary tests however show that the cost of using strict thresholds such that a high number of reduction loops are required have a huge impact on the time it takes to achieve the desired reduction.

[2] picasa.google.com

The thresholds used during experiments are based on this reasoning, with focus on accuracy rather than performance, but no scientific experiments have been performed to compare the effect on accuracy and performance of using different thresholds, since the specifics of the image filters are not the focus of this thesis. The results shown in both 6.5, 6.3 and 6.4 will be much affected by use of different transformation rule configurations, however the effect on accuracy should not change that much.

### 6.4.2 Effect of image metrics distribution

All in all, a vast number of image metrics are used in this system, either implicitly or directly. Most of the image and event metadata used, such as the situations (indoor, face, steering wheel etc.) and the image quality measure, are collected outside of this system in previous works, and have not been evaluated in this system. Some of them are nevertheless active for example when we evaluate the accuracy of presentation-time reduction: the situations queried for in the preceding query leading up to object presentation are used in the ranking of image data in the object, and will have an impact on performance that is not measured. The accuracy for detection of some of these situations are not very high, and we have seen that for instance in some events with only outdoor activity, almost all the image data have been marked as being indoor. Inspection of the images will however show the underlying data that have been used in the reduction loop algorithms, and it is the usage of this data that have been evaluated, not weather the underlying data reflects the human perception of the image.

For the quality measurements, there were no defined thresholds for what could be considered a "high" quality or "low" quality image, and neither is the accuracy of the quality measure known. Figure 6.2 clearly shows that the image quality measure is not a normalized value. This uneven distribution of image quality metrics means that when a specified reduction is required, it is unknown at which quality threshold the desired reduction goal can be met, and that even if profiling were done up front so that the system would know how many images that would be removed by each stepwise increment of the quality threshold, we would still not know what effect that reduction would have on the accuracy.

## 6.5 Summary

In Section 6.1 we identified our goals for our Lifelog library system to be performance and accuracy during data reduction, and coined our terms for success in achieving this. The dataset used during the experiments is introduced in 6.2, and a set of experiments is presented in 6.3, measuring data reduction, performance, and accuracy. In 6.4 the experiments and their results are evaluated in relation to the requirements specified in Chapter 3.

# Chapter 7

# Conclusion

In Chapter 1 we introduce Lifelogging as an interesting domain combining ideas from digital libraries and wearable capturing devices to create and archive a digital representation of the memories and experiences we receive during our everyday life. Ongoing research focus both on capturing, management, access interfaces, and how the use of a HDM system affects us. An increasing amount of personal multimedia content is captured, and for many people the media stored on our personal computers functions as a basic version of a lifelog digital library, but poorly adapted to the task.

While ideally we could want to store everything, a report from 2007 [1] warns us that storage space may not continue to be in abundance, and information retrieval from complex multimedia data only gets harder the more we store. None of the related work discussed in Chapter 2 focus on the issue of limited storage space, and normal approaches like data compression is already embedded into modern digital media format, leaving little room for improvement.

In our previous work [3] we proposed a framework for automatic data reduction that showed good experimental results. Based on the requirements in Chapter 3 for such a framework applied to the Lifelog domain, in Chapter 4 we proposed an extension to our previous framework so it would be better adapted to the harder requirements of the complex dataset in a Lifelog application. We created a prototype implementation of a digital library using our extended framework, with a web-based search interface as shown in Chapter 5. With this prototype we could now put our extended framework to the test and see if we would still be able to achieve data reduction and at the same time fulfill the requirements defined in Chapter 3. The experiments performed are described and evaluated in Chapter 6.

## 7.1 Achievements

The problem definition from 1.2 is stated below:

> *This thesis shall develop and evaluate in a scientific context a framework and prototype system for automatic reduction of data through filtering techniques. The specific application domain at study is related to the Lifelog project. The thesis shall focus on how to support a system enabling accurate data reduction, but retaining data quality.*

We have developed a framework and prototyped a system for automatic reduction of data through filtering techniques, applied to the Lifelog domain. The system was evaluated in a scientific context, and was found to enable accurate data reduction while retaining data quality.

In Chapter 6.4 we have evaluated our extended framework as implemented in the prototype. We show that we are able to perform the expected data reductions on the new domain of Lifelog data, and that a very high accuracy is achieved. Without the high accuracy, data quality could not be retained. We further show that we are able to support access from devices with different display capabilities, automatically adapting the representation given of the objects based on dynamic query-time information. Our system provides a new and unique way of reducing the data and improving object representation at query-time based on the access device and search input, and we showed that our system performed better than traditional representations.

In conclusion, this thesis has presented a framework and prototype for improving the usefulness of multimedia archives by applying automatic data reduction techniques.

Finally, some possible future work will now be presented.

## 7.2 Future work

In this thesis, transformation rules and their configuration have been mentioned at several occasions. In combination with the defined object state model for the digital library, they define the different ways objects can change state. It is however not easy to get a good understanding of this without thorough inspection of the different transformation rule configurations, and the digital library setup. It would be interesting to instead define a general state model that could be defined for example in XML format. State models can be easily visualized, and use of a central XML file would make reconfiguration of the system much easier.

Figure 7.1 show an example state diagram. This state diagram is not representative of the configuration used in our prototype implementation. From Import, objects have to go through a special Filter state before arriving in a Published state QP. The Filter state can for example stop intimate data to become published in the system. In a Published state, objects can change between Similarity

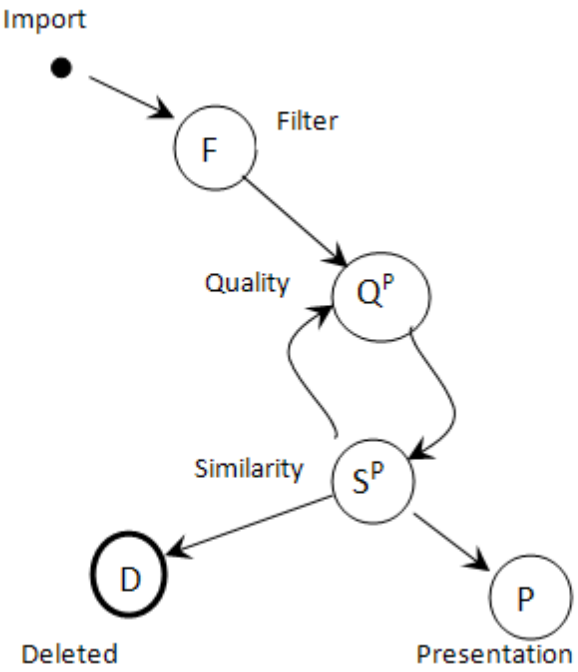



Figure 7.1: Proposed State Transition Diagram for Digital Library.

and Quality, which are example of two transformation rules. In the figure, an object can only be Deleted or Presented if it's in the Similarity state.

The dataset used in this work was captured in April of 2007, and as such it was not live data, and never changed. Also, it was of limited size. In future work it would be interesting to implement this framework for use on a large-scale system with live data, possibly captured from many users at once. Focus must then be added to performance, utilizing load-balancing and multi-threading.

Another issue not explored in this work is automatic use of context as input to the interface. For example some mobile devices have in-built GPS Receivers that can transmit the GPS coordinates of the user while accessing the lifelog library. Additional information can for example be a timestamp. Using this as input, a query could automatically be performed, returning objects from the lifelog library relevant to the current context of the user.

# Bibliography


[1] John F. Gantz, D. Reinsel, C. Chute, W. Schlichting, J. McArthur, S. Minton, I. Xheneti, A. Toncheva, A. Manfrediz. The Expanding Digital Universe. And IDC Whitepaper, 25. February 2007.

[2] L. Candela et al: The DELOS Digital Library Reference Model - Foundations for Digital Libraries. Version 0.98, February 2008

[3] T. Aarflot, C. Gurrin, D. Johansen. A Framework for Transient Objects in Digital Libraries. ICDIM 2008.

[4] J. Gemmell, G. Bell, R. Lueder, S. Drucker, C. Wong. MyLifeBits: Fulfilling the Memex Vision. Procedings of the tenth ACM International conference on Multimedia, 2002.

[5] Peter J. Denning. Computing as a discipline. Communications of the ACM Volume 32 , Issue 1, January 1989.

[6] J. Gemmell, L. Williams, K. Wood, R. Lueder, G. Bell. Passive Capture and Ensuing Issues for a Personal Lifetime Store. Proceedings of the the 1st ACM workshop on Continuous archival and retrieval of personal experiences, ICM 2004.

[7] A. R. Doherty, C, Ó Conaire, M. Blighe, A.F. Smeaton, E. Noel. Combining image descriptors to effectively retrieve events from visual lifelogs. MIR 2008 - ACM International Conference on Multimedia Information Retrieval, 30-31 October, Vancouver, Canada.

[8] D. Byrne, A. R. Doherty, C. G. M Snoek, G. J. F. Jones, A. F. Smeaton. Validating the detection of everyday concepts in visual lifelogs. Semantic Multimedia. Lecture Notes in Computer Science, 5392 . Springer Berlin / Heidelberg, pp. 15-30.

[9] A. Doherty, D. Byrne, A. F. Smeaton, G. Jones, M. Hughes. Investigating Keyframe Selection Methods in the Novel Domain of Passively Captured Visual Lifelogs. CIVR 2008 - ACM International Conference on Image and Video Retrieval, Niagara Falls, Canada.

[10] M. Blighe, H. le Borgne, N. O'Connor, A. F. Smeaton, G. Jones. Exploiting context information to aid landmark detection in SenseCam images. In ECHISE 2006 - 2nd International Workshop on Exploiting Context Histories in Smart Environments (Ubicomp 2006), Orange County, CA, 2006.

[11] T. Hori, K. Aizawa. Context-based video retrieval system for the life-log applications. Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval, Berkeley, California, USA, 2003.

[12] Bush, Vanneva. As We May Think. The Atlantic Monthly, 176(1), July 1945, 101-108.

[13] B. Gates, N. S. Myhrvold, P. Rinearson. The Road Ahead. Viking Penguin, New York, 1995.

[14] E. Freeman, D. Gelernter. LifeStreams: A storage model for personal data. ACM SGIMOD Bulletin 25, 1, March 1996, pp. 140-170.

[15] S. Mann, "Wearable Wireless Webcam," personal WWW page, http://wearcam.org (http://n1nlf-1.media.mit.edu), 1994.

[16] I.H. Witten, D. Bainbridge. How to Build a Digital Library. Morgan Kaufmann, San Francisco, CA, 2003.

[17] A. P. Bishop, S. L. Star. Social informatics for digital library use and infrastructure. Annual Review of Information Science and Technology, NJ, 1996.