UiB | Dept. of Information Science and Media Studies
UiTø | Dept. of Computer Science
NTNU | Dept. of Computer and Information Science
Telenor | Telenor R&I

CAIM
CONTEXT-AWARE IMAGE MANAGEMENT

Bergen By

a Multi-modal Image Database

Thor Steinar Møller

Sept. 2009

CAIM-UiB

TR-9

Acknowledgements
Many thanks to Lisa Fremmerlid for inviting me to Trondheim to photograph Nidarosdomen, and to Mikael Tjemsland and Mathias Hellevang for lending me their precious cameras. Without you image capturing would have been much harder.

Abstract

This report is based on the work on BergenBy database during the summer of 2009. It describes the new functionality that has been developed, how it is implemented and how the database is to be built if needed. At the end it also suggests some new ideas for functionality that could be useful to have in the future.

1.0 Project objective

The BergenBy database is located on bulmeurt.uib.no and serves as the database backbone for the prototypes developed by the CAIM project at the university of Bergen[1]. By June 2009 the database contained 250 images of 23 objects (buildings, statues etc.) in and around Bergen, Norway. The primary goal for this database project was to expand the image collection by increasing both the number of objects and images of those. It was also decided to include some drawings or drawing-like images to enable query by sketch through the prototypes. All images contained in the database would also be cut into 9 (3 x 3) tiles[2] to enable testing to see if it had an impact on precision / recall when searching for images in the database.

The secondary goal was to expand the database structure to include an audio file describing each object. All objects contained in the database had a text document describing the object, which would serve as a manuscript for the audio file either by manually reading, or by utilizing text-to-speech software.

Tertiary goals were to implement visual indexes to support visual queries, and text indexes to support general keyword / full text search[3].

[1] See http://caim.uib.no for complete list of prototypes
[2] See Carlson (2009) for java implementation specifications
[3] See Carlson (2009) for implementation specifications

1.1 Functionality:

By increasing the number of objects in the database one would implicitly increase the number of images, as it would be necessary to capture images of these objects. The following metadata for the objects was to be collected and stored in the database:

- Location for object
- Type of object
- Descriptive text
- Audio file describing the object

Increasing the number of images would give more precise results when evaluating the precision / recall for the prototypes. Approximately 250 images were to be captured and the following metadata were to be collected and stored in the database together with each image:

- Location of where the image was captured
- Time (time, date, year)
- Weather
- Objects in photo
- Activity depicted
- Keywords
- Photographer

In addition each object should have one or two historical images from the university library's collection of historical images. This proved to be difficult to fulfill for all objects due to the fact that some of the selected objects were too new to have existed in any historical context, i.e. "Den Blå Steinen" which was unveiled in March 1993. Lastly some of the images were to be converted to sketch like images using a software tool to support query by sketch in the prototypes.

In addition to containing full-scale images the database was also to contain tiles from the images. These were to be produced by cutting each image in the database into 9 pieces and linking these to the original image.

1.2 Database structure

The database structure builds directly on the database that existed from the work done in 2008, described in CAIM-UIB TR-5 (Langøy, 2008). After adding the new metadata requirements the following metadata were to be stored in the database:

- GPS based location[4]
- Timestamp
- Weather conditions in photo
- Number of primary objects in photo
- Name(s) of objects in photo
- Types(s) of objects
- Activity depicted
- Keyword
- Text
- Audio
- Tiles

There was also a need to develop procedures and functions to import the audio files and tiles, as well as building indexes for the images, tiles and texts to enable a more effective search.

2: Design –types & structure

Retrieving the necessary DDL for the database proved to be difficult due to the fact that the projects startup occurred on the same time as the Norwegian summer holiday. After finally retrieving them it became evident that the VISI2 and MMIR2 prototypes were running on two separate databases, which needed to be combined into one. The two prototypes also made use of different procedures for searching, and as it was a requirement that these prototypes were to be kept alive and fully functional the database had to sustain its

[4] Both camera and object location are stored and in some cases may vary with up to 1,1km in distance

structure as described in Langøy (2008). In other words the structure could be additions, but no alterations were made to the existing database structure.

3.0 Database structure and design

The two databases that supported MMIR2 and VISI2 were copied and merged into a new database so that it contained all the procedures and functions needed by both prototypes and could be used for testing without interrupting the live databases supporting the prototypes.

The final structure for the database is based on the data types listed in Table 1.

Table 1. Types in BergenByDb

IMAGE_TYPE	
ID	NUMBER
Caption	VARCHAR2(255)
Photographer	REF AUTHORTYPE
Texts	NT_TEXT_REF
Image	ORDSYS.ORDIMAGE
ImageSignature	ORDSYS.ORDIMAGESIGNATURE
Coordinates	MDSYS.SDO_GEOMETRY
ImgDate	TIMESTAMP
Weather	VARCHAR2(50)
Activity	VARCHAR2(255)
Keywords	NT_KEYWORDS_REF

IMAGE_TEMP_TYPE	
Id	NUMBER
Caption	VARCHAR2(255)
Image	ORDSYS.ORDIMAGE
ImageSignature	ORDSYS.ORDIMAGESIGANTURE
Coordinates	MDSYS.SDO_GEOGRAPHY
ImgDate	TIMESTAMP
Weather	VARCHAR2(255)
Activity	VARCHAR2(255)

OBJECT_TYPE	
Id	NUMBER
Name	VARCHAR2(255)
Age	NUMBER
Material	VARCHAR2(255)
Coordinates	MDSYS.SDO_GEOMETRY
Area	VARCHAR2(255)
Classification	VARCHAR2(255)
Texts	NT_TEXT_REF
Keywords	NT_KEYWORDS_REF
Audio	NT_AUDIO_REF

AUTHOR_TYPE	
Id	NUMBER
Name	VARCHAR2(255)
Affiliation	VARCHAR2(255)

CONTAINS_TYPE	
Object	REF OBJECT_TYPE
Image	REF IMAGE_TYPE
Location	CHAR(2)

KEYWORDS_TYPE	
Id	NUMBER
Word	VARCHAR2(255)

TEXT_TYPE	
Id	NUMBER
Summary	VARCHAR2(255)
Language	VARCHAR2(50)
Text	CLOB
Authors	NT_AUTHOR_REF

TILE_TYPE	
Id	NUMBER
TileNr	NUMBER
Tile	ORDSYS.ORDIMAGE
TileSignature	ORDSYS.ORDIMAGESIGNATURE

TILE_CONTAINS_TYPE	
Image	REF IMAGE_TYPE
Tile	REF TILE_TYPE

AUDIO_TYPE	
Id	NUMBER
Language	VARCHAR2(255)
Audio	ORDSYS.ORDAUDIO
Narrator	NT_AUTHOR_REF

Note that only TILE_TYPE, TILE_CONTAINS_TYPE AND AUDIO_TYPE are new types, and are explained in details below. The OBJECT_TYPE was extended during the project and the new version is also described in details. All other

types existed in the previous BergenBy Db and detailed explanation is omitted in this report[5].

3.1 In depth description of new types

TILE_TYPE: A fraction of an image in the database

Id = A unique number to identify a particular tile

TileNr =A number describing what part of the source image the tile relates to. All tileNrs start in upper left corner of the main image so the upper left tile is numbered 1. The lower, right tile is numbered with the maximum number of fractions the main image is divided in to, i.e. 9 tiles would number the lower right tile with 9.

Tile = The tile itself.

TileSiganture = A visual signature used for indexing searching in tiles

TILE_CONTAINS_TYPE: Relates 9 tiles to one main image[6].

Tile = A reference to a tile

Image = A reference to an image

AUDIO_TYPE: The audio files in the database

Id = Unique number to identify a particular audio file

Language = The language of the audiofile

Audio = The audio file itself

Narrator = A nested table with references to author-types narrating the audio file

[5] See CAIM-UIB TR-5 (Langøy, 2008) for details.
[6] See CAIM-TR-8 (Carlson, 2009) for visual example of tiles.

OBJECT_TYPE: The objects in the database

Id = A unique number to identify a particular object

Name = The name of the object

Age = Year of construction / unveiled

Material = Building material of object

Coordinates = GPS coordinates of object

Area = geographic area of object

Classification = Type of object, e.g. “statue”, “building”, or “church”

Texts = Nested table with references to texts describing the object

Keywords = Nested table with references keywords describing the object

Audio = Nested table with references to audio files describing the object

3.2 Data model for BergenByDb

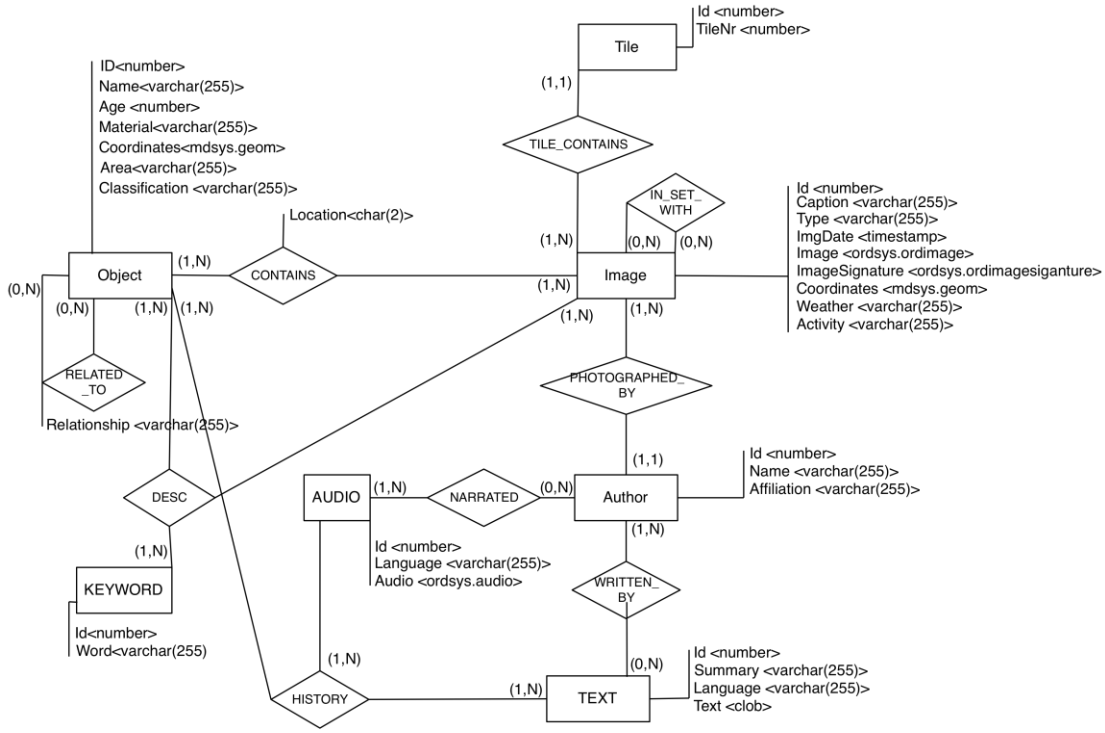

Figure 1. Data model for BergenByDb

4.0 Object selection, image capturing and audio file creation

To expand the database it was necessary to include both more objects and images related to these. Database expansion was the primary database objective for the 2009 project and the first task to be performed.

4.1 Object selection

All new objects initially had to fulfill certain criteria with regards to location, but all objects did not meet these. Initially all new objects were to be located within 50 meters of an existing object, but given Bergen's geography that was not always possible. Clusters of objects should not exceed five objects, and this was fulfilled for all objects. For existing objects in the database see (Langøy, 2008). The objects selected were categorized according to a pattern similar to the one used by the 2008 project. All objects are instances of 1 or more of the following categories: Churches, statues, buildings, museums, monuments, culture, fountain, boat and places. Due to the fact that an object may very well be of more than one category more than one keyword has sometimes used, i.e. a museum might also be annotated with the keyword "building" in addition to "museum".

The new objects chosen are believed to be of interest to tourists, but contrary to the objects selected in 2007 and 2008 some of the objects from 2009 are of newer date. This is based on the thought that they might appeal to a younger audience than the mere historical objects. By including objects of younger age it was also easier to locate objects in closer proximity to the objects that already existed in the database, thus complying with the demand that all new objects should be within 50meters of an existing object. One particular object that does not comply with this requirement is Nidarosdomen, a cathedral located in the city of Trondheim, located 630km from Bergen. This object was added because of the CAIM project[7] relationship between UiB in Bergen and NTNU in Trondheim.

[7] See project members at http://caim.uib.no

As of Sept.1, 2009, the BergenBy DB contained the following 47 objects:

Churches: Domkirken, Johanneskirken, Korskirken, Fantoft Stavkirke, Nykirken, Mariakirken and Nidarosdomen.

Statues: Christian Michelsen, Edvard Grieg, Ludvig Holbergn, Henrik Ibsen, Amalie Skram, Nordahl Grieg, Ole Bull and Snorre Sturlasson.

Buildings: Børsbygningen, Grieghallen, Zachariasbryggen, Gamlehaugen, Akvariet, Stadsporten, Den Nationale Scene and Nedre Fløibanestasjon.

Museums: Vestlandske Kunstindustrimuseum, Rosenkrantztårnet, Håkonshallen, Troldhaugen, Bergen Kunsthall, Naturhistorisk Museum, Kulturhistorisk Museum, and Bergen Kunstmuseum Lysverket.

Monuments: Sæverudsmonumentet, Musikkpaviljongen, Sjømannsmonumentet, Rogaland Cross, Den Blå Stein, Liggende poet, Totempælen, Heksesteinen, and Petter Dass & Dorothe Engelbretsdotter memorial.

Places: Festplassen, Lille Lungegårdsvann, Fløien and Bryggen

Boats: Statsraad Lehmkuhl and Beffen

4.2 Image capturing

As far as possible, new objects were to have images according to the following requirements:

- Five full views of the object
- Three partial views, for instance close-up of object details
- Multiple object images, meaning that objects had to be within viewing distance from each other
- Long-distance (> 200m) images, meaning that there had to exist a free line of sight from the point of photographing to the object
- Different weather conditions, meaning that all images could not be captured at once

In addition to the new images, I wanted to capture some new images of existing objects, especially during low light conditions, such as evening and night. I also wanted to capture long distance images using a telescope lens so that the object would fill most of the image, but the position of where the image was to be captured from would not be the same as the position of the object depicted. The reason for this was to see how much the CBIR-search is affected by the GPS variable.

Image capturing could have been performed using the existing MMIR2 prototype, but image capturing would have been very time consuming due to the fact that the MMIR2 software runs slowly on a mobile phone. Based on experiences from the 2008 project[8] it was decided to use a standalone camera to capture images. In practice 3 DSLR[9] cameras were used. In addition to the images captured I searched my private photo library and discovered images of some of the objects. These images included an image of Grieghallen in snowy conditions, an impossible image to capture in Norway during summer, and images of Nidarosdomen.

Based on the documentation from the 2008 project the metadata editor had problems with large images. To circumvent future problems using this editor all images captured were scaled down in a batch process using Canon's software Digital Photo Professional. During this processing all images were also given file names based on the main object depicted with a trailing number starting at 00 incrementing by 1,eg. "bryggen.01", to ease file handling. After resizing and renaming, the files were uploaded to bulmeurt.uib.no for storage in the database. All image files have to be located on a server to be inserted into the database because of the image import procedure [10].

The current version of BergenBy DB contains ca. 500 images of the 47 objects. The full listing of images can be found at http://bulmeurt.uib.no:8500/caim/ImageCatalog/.

.

[8] Langøy (2008) states that the 2008 project used a standalone camera for image capturing.

[9] Digital Single-Lense Reflex camera

[10] See (Langøy,2008) for HTTP_IMPORT_IMAGE implementation details.

4.3 Audio file creation

Initially all audio files were to be created using some sort of text-to-speech software. Most of the software systems with this functionality are English, but a lot of the objects have Norwegian names, hence making use of text-to-speech software difficult. Instead of modifying the text-to-speech software it was decided that it was quicker to read the texts aloud and record the sound. All texts were recorded using the built in Macbook microphone and Audacity sound recording software. After recording, the files were compressed to mono track mp3 using the LAME mp3-encoding library. Compressing the audio files was not absolutely necessary, but it decreased the time needed to download them to the MMIR3 prototype as this occasionally makes use of 3G / EDGE internet connections[11].

5.0 Tools used in the project

The tools used for the database section of the 2009 CAIM project were:

-MacBook (rev 2.1) laptop running Mac OS X ver. 10.5.7

-Sun Virtualbox running Windows 7 beta release

-Oracle SQL Developer OS X ver. 1.5.4

-Google Maps

-Google Earth

-Canon 450d dSLR camera with 18-55mm lense

-Nikon d40 dSLR camera with Nikkor 55-200mm telelens

-Pentax K200d dSLR camera with 200mm Tamron telelens combined with a 2X converter (400mm effective)

-Canon Digital Photo Professional photo editing software

-TextEdit ver. 1.5

-Audacity ver. 1.2.5 sound recording and editing software

-Cyberduck ver 3.0.2 FTP client

[11] 3G allows simultaneous use of speech and data services and data rates up to 14.4 Mbit/s on the downlink and 5.8 Mbit/s on the uplink and large files take a long time to download.

The MacBook running Mac OS X was the primary operating environment, but due to the fact that some of the features, such as automatic duplicating of databases through the OS X version of Oracle SQL developer was not working, Sun Virtualbox was used to run a virtual Windows 7 environment on the MacBook.

Oracle SQL developer was used to write SQL for creation and alteration of BergenByDb components. All inserts and other PL/SQL statements were stored by copying them to TextEdit and saving them in plain text format (.txt), thus enabling a quick, platform independent way of reading through the PL/SQL statements.

The three digital single-mirror reflex cameras were used for image capturing. The primary camera was the Canon 450d, but due to the lack of 200mm lens for this camera a Nikon d40 was used. To be able to capture images up to 1,1km away from the object a 400mm telelens was constructed by combining a regular 200mm telelens with a 2X converter mounted on a Pentax K200d. The reason for using a third camera was because the 2X converter had a Pentax K-mount bayonet which did not fit on either of the two first cameras.

All images captured were much larger than what one would expect from a mobile phone due to the fact that a dSLR camera has a larger image sensor than a mobile phone. Because of this the Digital Photo Professional software from Canon was used to scale down the images so that file size was more similar to what one would expect from a mobile phone.

Audacity, an open source audio recording and editing software was used to record and edit the audio files describing the objects.

Cyberduck FTP client was used to upload all files to bulmeurt.uib.no. It was also used for directory creation on bulmeurt.uib.no.

6.0 New Db functionality

To comply with the project objectives the database had to be extended somewhat both to enable storing of new data, such as audio files, but new procedures were also needed, such as those responsible for importing tiles and audio files.

6.1 Image signature indexes

To enable quicker image search all image signatures, for both images and tiles, were to be indexed using the Oracle functionality for this. Image indexes make use of their own tablespace and this needed to be created in the database before the indexes were created. Unlike the rest of the database structure, like tables, functions and procedures, which are deleted, when dropped, table spaces are persistent so it was only necessary to create them once. The image signature index, named IMAGE_IDX, resides at *ordimage_idx_tbs_1* while the index for tile signatures, TILE_IDX, resides at *ordimage_idx_tbs_2.* Both indexes have data files with names corresponding to the tablespace names.

It is advised that all indexes are to be dropped before importing new images and rebuilt after, rather than updating the indexes as images are imported. This is due to performance limitations in the index that would make image imports time consuming. During the project the validity of this advice was not tested, but as it is an Oracle advice I felt it should be followed.

6.2 Tile importing

All ordinary images can be cut into a number of tiles, thus enabling search on parts of the images in the database. This is done by the "tilebuilder" which is a java servlet located on bulmeurt.uib.no[12]. This servlet makes use of two procedures; HTTP_IMPORT_TILE and UPDATE_TILE_CONTAINS. HTTP_IMPORT_TILE is built on the principles of HTTP_IMPORT_IMAGE, but takes fewer parameters as a tile is merely a section of an image and it is impossible to know if there are any objects located on that particular tile.

The parameters for the function are:

p_URL varchar2, -- URL of folder containing the image file
p_Filename varchar2, -- Filename of image file
p_TileNr number, -- Describing where in the image the tile belongs
p_Results OUT NOCOPY VARCHAR2 --return string to the servlet

In addition to these parameters all tiles that are imported get a unique ID from a sequence which increments by one as tiles are imported.

The UPDATE_TILE_CONTAINS is responsible for updating the table TILE_CONTAINS, where the relations between the ordinary images and the tiles are saved. All tiles have a reference to the initial image so that a CBIR-search result on a tile can be linked to an ordinary image.
The UPDATE_TILE_CONTAINS procedure takes the following parameters:

p_ImageId number, -- A Image ID
p_TileId number, -- A Tileid
p_result OUT NOCOPY VARCHAR2 --A result string to the servlet

[12] See VISI3 technical report (Carlson, 2009) for the java implementation specifications.

6.3 Searching in tiles

The MMIR3 prototype has the functionality[13] to search with a selection of an image. When this is done the search does not search in the ordinary images, but on tiles based on parts[14] of the ordinary images. The package containing the procedures for this is named TILE_SEARCH. The design of these procedures bare close resemblance to the design of the normal CBIR-search, except it has fewer parameters.

6.4 Audio importing

The MMIR3 prototype has functionality that enables it to play audio files which describe a given object. These audio files are stored in a nested table connected to the object they belong to. All audio files have to be imported into the database before they can be retrieved by the MMIR3 prototype, and like the images the audio files have to be located on a web server before being imported. The procedure responsible for audio import is named HTTP_IMPORT_AUDIO, and is built on the same pattern as the image import procedure except that it saves the audio as ORDSYS.AUDIO.

HTTP_IMPORT_AUDIO takes the following parameters:

p_URL varchar2, -- URL of folder containing the audio file

p_Filename varchar2, -- Filename of audio file

p_Language varchar2, -- Describing the language of the spoken words

p_Narrator number --A reference to an instance in of the Author type

6.5 Retrieving audio references

The MMIR3 prototype needs to retrieve the audio files before playing them. This is handled by the GET_AUDIO_REF function, which takes a given audio-ID as parameter and returns the corresponding reference. The audio-ID is stored as a parameter in the object a user wishes to retrieve an audio-file for so the function

[13] See CAIM-TR-7 (Hellevang, 2009) for the java implementation specifications
[14] The default number is 9 tiles pr image, but this can be altered as a parameter in the tilebuilder. See CAIM-TR-8 (Carlson, 2009) for details.

does not take an object-ID as a parameter as the MMIR3 prototype already knows which object the audio file relates to.

6.6 Retriving tile references

To retrieve a given tile the function GET_TILE_REF can be called. The function takes a tile-ID[15] and returns the corresponding tile.

6.7 Designation of audio- and tile-IDs

There exist two sequences to handle ID designation for audio files and tiles as they are imported into the database. These are named AUDIOS_SEQ and TILE_SEQ. The sequence is incremented by one as files are imported and designate IDs accordingly.

7.0 Instructions to build db from PL/SQL repository

To build the database from scratch a certain order has to be followed to avoid errors in the database structure, as well as in the data structure. All inserts are located in the folder BuildDB located on the CD following the paper edition of this report, and have to be run in Oracle SQL Developer in the following sequence:

1. typesAll.txt
2. tablesAll.txt
3. sequencesAll.txt
4. createTablespaces.txt[16]
5. functionsAll.txt
6. packages.txt
7. CAIM_BERGENBY.txt
8. VISI_BERGENBY.txt
9. TILE_SEARCH.txt

[15] Must NOT be confused with TileNr. Only Tile-IDs are unique identificators.
[16] This is only needed on a clean Oracle database, if the bergenByDb has been built before these tablespaces for the indexes should exist and an error will be raised by Oracle if creation is attempted

The three last packages above consist of a package head and a body. The heads have to be imported separately before the bodies are imported, otherwise Oracle will raise an error.

10. createIndex.txt[17]

After the structure is established the data can be inserted by running the inserts located in the data folder on the CD flowing the paper edition of this report. All inserts should be run in the following sequence to avoid errors:

1. authorsAll.txt
2. keywordsAll.txt
3. audios2009.txt
4. images.txt
5. images2008.txt
6. images2009.txt
7. sketches.txt
8. textsAll.txt[18]
9. objects.txt
10. objects2008.txt
11. objects2009.txt
12. contains.txt
13. contains2008.txt
14. contains2009.txt
15. sketchContains.txt

The location of the images and audiofiles can be read as the first parameter from the insert statements in images.txt, iages2008.txt, images2009.txt and sketches.txt.

[17] It is advised by Oracle that image indexes are built after images are inserted due to performance limitations.
[18] The text index has to be updated after any new text is inserted. At the end of the file textsAll.txt there is a PL/SQL statement that takes care of this.

8.0 Future development ideas

The following sections include future development ideas based on thoughts that occurred during the 2009 project. The features may not be implemented in the future, they are merely ideas for a future version of BergenByDb, although some of them seem more natural to implement than others.

8.1 Indexing GPS locations

Indexing of GPS locations was not completed during the 2009 project, but should be implemented. Oracle has built in functionality to do this so it should be a rather straightforward task to implement.

8.2 Marking object positions on images

The CONTAINS-table has, in addition to references to images and objects, a column for storing the location of an object in a specific image, eg. If an object occurs in the very center of an image the appropriate variable to store would be "C", or if the object occurs in the upper left corner the variable to store would be "NW"[19]. In the section for future development ideas in the VISI3 report it is outlined the possibilities for a CMS[20] for the images were users could mark objects on images. The functionality to store object location on images would be useful for this.

8.3 Low scale images as metadata

Early in the 2009 project it was experimented with reducing the number of colors on the images in the database. The results of these experiments were inconclusive, but in the future it might be useful to include a low scale image with fewer unique colors, e.g. ten colors pr image as a separate metadata instance for each image. The rationale for this is that if a user draws an image to use in CBIR search, using either the MMIR3 or the VISI2 / 3 prototype a user would not use more than a few colors, were as most of the images in the database consists of many thousands of colors pr image.

---

[19] Location markers follow the pattern: upper middle = "N". Lower middle = "S". Midle left side = "W". Midle right side = "E". Combinations of these can be used to map objects in corners.

[20] See CAIM-TR-8 (Carlson, 2009) for details.

9.0 References

Carlson, Christoph. (Sept. 2009). *VISI3 – Context Aware Image Retrieval*. CAIM-TR 8. Dept. of Information and Media Sciences, Univ. of Bergen.

Hellevang, Mathias (Sept. 2009). *MMIR3 – Mobile Multimedia Image Retrieval.* CAIM-TR 7. Dept. of Information and Media Sciences, Univ. of Bergen

Langøy,M. (Aug.2008) *BergenBy Database & Metadata Editor.* CAIM-TR-5, Dept. of Information and Media Sciences, Univ. of Bergen.

Appendix A

The image catalogue is too big to be included in this report. It can be downloaded separately from: http://bulmeurt.uib.no:8500/caim/ImageCatalog/ NB! Address is case sensitive.

Appendix B

CAIM database source code is restricted. Access to database statements can be granted by CAIM project administrator Joan Nordbotten