# Using PhotoCube as an Extensible Demonstration Platform for Advanced Image Analysis Techniques

Grímur Tómasson, Hlynur Sigurþórsson, Kristjan Runarsson, Gisli Kristjan Olafsson, Björn Þór Jónsson, Laurent Amsaleg

# Using PhotoCube as an Extensible Demonstration Platform for Advanced Image Analysis Techniques

Grímur Tómasson[†] Hlynur Sigurþórsson[†] Kristján Rúnarsson[†]
Gísli Kristján Ólafsson[†] Björn Þór Jónsson[†] Laurent Amsaleg[§]
[†]School of Computer Science
Reykjavik University, Iceland
bjorn@ru.is
[§]IRISA–CNRS
Rennes, France
laurent.amsaleg@irisa.fr

## Abstract

*As digital image collections have been growing ever larger, the multimedia community has put emphasis on methods for image content analysis and presentation. To facilitate extensive user studies of these methods, a single platform is needed that can uniformly incorporate all the analysis and presentation methods under study. Due to its extensibility features, a plug-in API for image analysis methods and a browsing mode API for presentation methods, we believe that the PhotoCube browser can be that platform. We propose a demonstration focusing primarily on these features, allowing participants to appreciate the full potential of PhotoCube as a demonstration platform.*

## 1 Introduction

Since the introduction of personal computers, and digital cameras in particular, collections of digital images have been growing ever larger. Personal collections now often contains tens of thousands of images, while professional collections and web-scale collections are much larger.

In the multimedia community, there has been a corresponding emphasis on methods to analyze images to enable various usage scenarios. These analysis methods include face recognition [12], object recognition [4], timeline analysis [3], and many more. There has also been significant emphasis on various methods for presentation of images, such as the image 3D image walls, photo tourism [7], and the compaction of images onto the screen [1].

### 1.1 Need for a Demonstration Platform

With the proliferation of analysis and display methods, it becomes extremely important to conduct user studies to compare the methods and analyse their impact on user productivity and satisfaction. So far, however, such efforts are rare and typically underwhelming. It appears that there are two main reasons for this lack of user studies.

First, running proper user studies is very time-consuming and expensive. There is usually neither time nor funding for such efforts. But the second issue, which we address here, is that even when there is time or funding, it is difficult to avoid comparing apples with oranges. Each analysis method or presentation method is typically embedded in a particular system produced by the authors, with a specific user interface and specific visual features, and this diversity makes an objective comparison very difficult.

What is needed is a single platform that can uniformly incorporate all the analysis and presentation methods under study. For the last two years we have been developing the (soon-to-be open source) PhotoCube browser, which we believe can become such a platform.

### 1.2 The PhotoCube Prototype

PhotoCube is a three-dimensional graphical user interface for effective image browsing [11, 10]. PhotoCube implements a novel multi-dimensional data model for media browsing, called ObjectCube, which in turn is based on the multi-dimensional analysis model that has been so successfully employed by the business intelligence community for various data warehousing and OLAP applications [2].

The two key entities of the data model are *tag-sets*, which encapsulate all tags describing a particular concept that a user might be interested in, and *hierarchies*, which provide structure and organization to the tag-sets. A typical tag-set is *People*, which might have the hierarchies *Family* and *Friends*. Another typical tag-set is *Image Date*, which might have the *Year-Month-Day* and *Weekday* hierarchies.

These simple concepts are used to form media hypercubes, which represent the *browsing state* at each time. Typ-

ical browsing operations include drilling-down into a particular hierarchy, selecting tag-sets or hierarchies for the 3D display, and applying various filters to narrow the scope of the browsing state. While the multi-dimensional data model enables many interesting browsing scenarios, however, it is actually the architecture of the system that makes it very suitable as a general demonstration platform.

First, the underlying media server has a simple and very extensible *plug-in API*, which can be used to encapsulate image analysis methods. Upon insertion, each image is analysed by all the available plug-ins, which in turn generate tags into the corresponding tag-sets. As a strong case in point, we have recently integrated a face recognition plug-in by adapting an existing holistic face recognition algorithm to the plug-in API [5]. Since face recognition is among the most complex analysis methods as yet proposed, we believe that the plug-in API can be used to encapsulate any image analysis methods from the literature.

Second, the GUI is implemented using so-called *browsing modes*, which are applied to the (entire or partial) browsing state to decide the display method. The browsing modes are, again, encapsulated using a simple, yet extensible *browsing mode API*, which can be used to integrate multiple presentation methods into a single interface.

### 1.3 The Demonstration Proposal

We have previously demonstrated PhotoCube at ICMR 2011 where we focused on the browsing interface and operations. It was indeed during the discussions with ICMR conference attendees that the concept of a demonstration platform arose, as some attendees pointed out the value of the system for demonstrating their work; they would not need to develop an entire browser, but could simply adapt their methods to the plug-in API and thereby gain access to all the existing browsing functionality.

At CBMI 2012, our demonstration will focus primarily on the plug-in and browsing mode APIs. We will bring a poster explaining the two APIs in detail, as well as the existing plug-ins and browsing modes. We will then dynamically insert images taken at CBMI—or supplied by participants—into PhotoCube and demonstrate the outcome of the image analysis plug-ins using the various browsing modes. The code will be available for review for those curious about the difficulty of integration. We believe that this demonstration will allow participants to fully appreciate the potential of PhotoCube as a demonstration platform.

The remainder of this paper is organized as follows. First, we briefly describe the ObjectCube data model and the media server implementing it, before describing the plug-in API. Then we briefly describe the PhotoCube interface, before focusing on the browsing mode API. Finally, we conclude with a summary of the demonstration.

## 2 ObjectCube

ObjectCube is a generic multi-dimensional data model[1] and a browsing engine based on the concepts of the well known multi-dimensional analysis [2]. In the ObjectCube data model, browsing operations are used to zoom into the media collection and construct multi-dimensional browsing sets, or cubes, similar to OLAP cubes but with objects instead of numerical facts. These cubes are then displayed using the PhotoCube interface, described next.

The ObjectCube model is formally defined in [8, 9], but in the following we review the model and the browsing engine architecture, focusing on the core aspects that are necessary for understanding the PhotoCube prototype.

### 2.1 The Data Model

The core concepts of ObjectCube are objects, tags, tag-sets, hierarchies, and filters. These concepts are used to construct multi-dimensional browsing cubes presented to users.

The first core concept of ObjectCube is the *object* which is an item of interest to the user (typically a photo). The second core concept is the *tag* which is any meta-data attached to an object. Objects and tags can be linked thanks to the following four multi-dimensional concepts derived from multi-dimensional analysis (MDA). Tags can be grouped in *tag-sets*. A tag-set is a set of tags that the user perceives to be related, typically a category of tags representing a notion that is important to the user. A *hierarchy* is a tree adding structure and order to a subset of the tags of a tag-set. More informally, it serves to categorize some of the tags of a tag-set. A *hyper-cube* is created by selecting and storing information about one or more tag-sets, or hierarchies, which the user wishes to browse objects by. Last, a *cell* is the intersection of a single tag from each of the dimensions (tag-sets or hierarchies) in a hypercube. The hierarchy, hyper-cube and cell concepts are very similar to the corresponding concepts in MDA; the main difference is that in MDA, the simple numerical facts are easily aggregated at higher levels in hierarchies, while in ObjectCube aggregation is an important research topic.

Data retrieval concepts must be defined together with the data model. The key retrieval concept is a *filter*, which is a constraint describing a sub-set of objects that the user wishes to browse. A filter can be applied to any dimension. Filters can be applied to single tags, can do range-filtering of tag values and can be applied to the nodes of the hierarchies of the data model. Applying filters results in a *browsing state* which is essentially a hyper-cube that stores the various predicates and information associated with the current set of media objects that are displayed to the user.

[1] Although we focus on image browsing here, ObjectCube is a general model which is not bound to a specific media type.

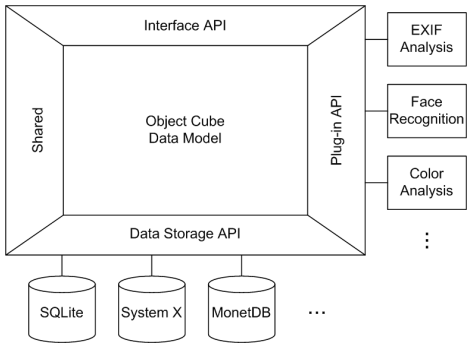


Figure 1: Architecture of the ObjectCube prototype.

### 2.2 ObjectCube Architecture

Figure 1 shows the architecture of the ObjectCube prototype. The prototype consists of a central logic module implementing the data model, as well as APIs for data storage, user interface development, and automated media analysis using the plug-in architecture. Plug-ins are software components which are called upon to analyze media objects during insertion and generate new tags or attach objects to existing tags. Currently, three plug-ins are implemented that: extract the EXIF meta-data associated with media files; extract faces from photos; and analyze the color composition of photos.

The ObjectCube prototype is written in C++ and makes extensive use of its standard library, as well as the TR1 C++ library extensions. The current implementation, without plugins, consists of approximately fourty thousand lines, and runs on Mac OS X 10.6 and Ubuntu Linux 10.04. Data-store implementations exist for three different relational database systems including MonetDB, an open-source column-store from CWI, that is known for its focus on performance.

We ran extensive performance evaluations checking the scalability of simple tag filtering, range filtering as well as hierarchical filtering. We measured the response time of the server when retrieving data sets having varying cardinalities. It basically shows that the ObjectCube is indeed usable in practice as it returns answers very rapidly. Detailed performance results are reported in [9].

### 2.3 The Plug-In API

Each plug-in implements a C++ class containing five functions. In our experience, these five simple functions are sufficient to implement even a complicated analysis method such as face recognition.

The goal of the `Process(photo)` call is to associate generated tags (new or previously generated) with bounding boxes; e.g., associating names with faces. Each bounding box may be associated with a set of tags, then considered proposals and ordered with the most likely tag first.

Subsequently, the user can `Confirm()` one of these tags, or even an entirely different tag if the plug-in fails include the correct tag in the proposed list. If the user adds a bounding box and associates it with a tag, this can also be done using the `Confirm()` call; the plug-in may choose to analyse such false negatives and learn from them. The user may also `Delete()` the bounding box if a) it is a false positive, or b) the user simply wishes to remove the tagging. Note that unused tags are garbage collected.

The user can `Rename(oldTag,newTag)` a tag, e.g., to change the name generated by the plug-in. Finally, the user may discover that two tags are actually the same and `Merge(tag1,tag2)` them.

## 3 The PhotoCube Prototype

PhotoCube [6, 10, 11] is the prototype photo browser we implemented on top of ObjectCube. It is strongly inspired by the browsers defined in the OLAP world, but also incorporates elements from 3D game-play. The GUI of PhotoCube supports three browsing dimensions, some or all of which can be used simultaneously. Since the underlying data model supports pivoting dimensions on and off the screen, this results in practical multi-dimensional image browsing. PhotoCube fully uses the ObjectCube model and can thus provide a rich set of browsing actions, such as drilling down through tag hierarchies and applying various filters to dimensions to focus on particular sets of images. It includes modules for pre-loading images and caching pictures in memory for efficiency.

### 3.1 The Browsing Mode API

The browsing mode API offers four interfaces. The `Initialize()` call loads any resources that may be needed; the `Load(c)` displays the images in sub-cube `c` and enables user interaction with the mode; finally, the `Disable()` releases all resources. While this API looks extremely simple, the complexity is actually hidden in the hyper-cube data structure `c` that is passed to the mode. This data structure initially encapsulates the browsing state that is returned from the media server. Subsequently, the user may choose a sub-set of the browsing state and switch to a different browsing mode, this subset is then encapsulated in the sub-cube `c` that is passed to the `Display()` call.

### 3.2 Current Browsing Modes

The default browsing mode is *cube mode*, which is a direct 3D representation of the ObjectCube data model; see

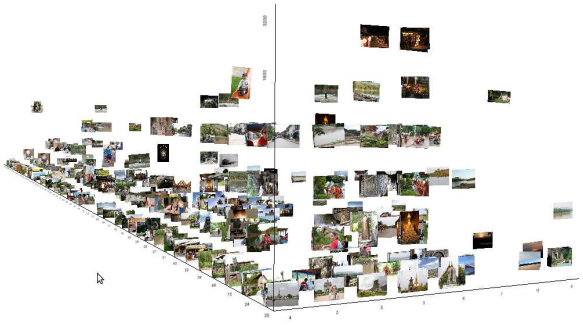

Figure 2: Screenshot of PhotoCube in cube mode.

Figure 3: Screenshot of PhotoCube in card mode.

Figure 2. In this mode, users can build an image hypercube by applying all the features supported by ObjectCube, such as filtering, pivoting, drill-down and roll-up. More details of the implementation of the cube mode, including the placement of image cells, may be found in [6].

The *card mode* is used to view details of images from selected cells in more details; see Figure 3. The images are presented as a row of standing cards, easily browsed.

As a proof of concept, we also implemented *shooter mode*, a simple arcade-style game where the selected images are used as shooting targets. Further modes could include various presentation modes from the literature, slideshows, web publishing and other such features.

### 3.3 PhotoCube Effectiveness

A preliminary user evaluation of PhotoCube, focusing on the browsing model rather than the user interface itself, has been performed. The users were five males, all with a computer science background but a varying experience with image browsers. The advanced user sample was chosen as more experienced computer users are more likely to understand the difference between the model and the user interface. The key results were that our participants found the prototype rather complicated and cumbersome to use, while finding it at the same time highly enjoyable, imaginative, and useful. The somewhat negative view of the prototype is not unexpected as our focus was on getting the most amount of features into it, at the cost of a poorer user interface. The evaluation results indicate, however, that the model has the potential to improve the state-of-the-art in image browsing.

## 4 Demonstration Summary

As mentioned in the introduction, our demonstration will focus primarily on the functionality and features of the plug-in API and browsing mode API. In this proposal we have described these APIs as well as the plug-ins and browsing modes that exist at this time; we also plan to implement further analysis plug-ins and browsing modes. For the demonstration, we will bring a poster explaining the two APIs in detail, as well as the existing plug-ins and browsing modes. We will then dynamically insert images taken at CBMI—or supplied by conference participants—into PhotoCube and demonstrate the outcome of the image analysis plug-ins using the various browsing modes. The demonstration will be very interactive, as conference participants will be able to use PhotoCube themselves. The code will also be available for review for those curious about the difficulty of integration. We believe that this demonstration will allow participants to fully appreciate the potential of PhotoCube as a demonstration platform.

## References


[1] B. B. Bederson. PhotoMesa: A zoomable image browser using quantum treemaps and bubblemaps. In *UIST*, 2001.

[2] R. Connelly and R. McNeill. *The Multidimensional Manager*. Cognos, 1999.

[3] S. Harada, M. Naaman, Y. J. Song, Q. Wang, and A. Paepcke. Lost in memories: Interacting with large photo collections on PDAs. In *JCDL*, 2004.

[4] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2), 2004.

[5] K. Rúnarsson. A face recognition plug-in for the PhotoCube browser. Master's thesis, Reykjavik University, 2011.

[6] H. Sigurthórsson. PhotoCube: Multi-dimensional image browsing. Master's thesis, Reykjavik University, 2011.

[7] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3D. In *SIGGRAPH*, 2006.

[8] G. Tómasson. ObjectCube – A generic multi-dimensional model for media browsing. Master's thesis, Reykjavík University, 2011.

[9] G. Tómasson, B. T. Jónsson, and L. Amsaleg. Objectcube: A novel data model for multi-dimensional media browsing. Technical Report RUTR-CS12002, Reykjavik University, 2012.

[10] G. Tómasson, H. Sigurthórsson, B. T. Jónsson, and L. Amsaleg. Photocube: Effective and efficient multi-dimensional browsing of personal photo collections. In *ICMR*, 2011.

[11] G. Tómasson, H. Sigurthórsson, B. T. Jónsson, and L. Amsaleg. Photocube: Towards multi-dimensional image browsing. Technical Report RUTR-CS12001, Reykjavik University, 2012.

[12] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35, 2003.