# A Client-Server Tool for Image Retrieval Applications

Pyry Kuusi, Kjell Lemström*, Petri Kanerva, Sami Koponen,
Sari Männynsalo, Juha Riihimäki, Janne Teinilä, Timo Virtanen

*University of Helsinki, Finland*

## 1 Introduction

The number of different image retrieval algorithms based directly on the features of a query image has drastically increased during the last few years because of the considerable research effort directed to this particular area. However, until these days there have not been any freely available platforms for these kinds of algorithms. The lack of such a system has led the researchers to create systems on their own just to test their algorithms. Since the algorithms have been implemented in different systems the comparison of the algorithms has also been difficult.

The starting point for our image retrieval project was to create a system to help the researchers working in the area of image retrieval by offering a system where the researchers can easily embed their algorithms without the bother of creating a whole system. We created a system called *PICSearch* [4], which is extendible, portable and freely available. It provides also an easy-to-use retrieving tool for normal use (but, so far, without too many retrieval algorithms embedded into it). There are various other image retrieval systems, such as *QBIC* [2], *Virage* [1], *MARS* [5] and $I^2C$ [6], but all these systems involve at least one component that is not freely distributed. To our knowledge, our system is the only one that is, as a whole, freely available.

The first version of *PICSearch* could retrieve images only from a local database. Since there are a lot of databases on the Internet, the obvious and desired extension of the system was to make the queries work over the Internet. We are now finishing a project whose aim is to implement a client-server extension to the *PICSearch* system. In the sequel, we refer to the new version of the system by using name *NetPICSearch*.

The usage of *NetPICSearch* has been made as easy as possible; the same program acts as a server or as a client depending on the situation. The client executes queries by forwarding the requests of the user to the server. Thus, the database accessing done by the server is completely hidden from the client. At connection time the server may advertise itself by proving a HTTP address that is to be used by the client (if the client has a WWW browser). Otherwise, the communication between the two consists solely of messages needed to run image queries.

We have tried to make the communication rate between the two ends as low as possible to maintain efficiency; normally only thumbnails of matched images are retrieved. The images in databases of servers are accompanied by algorithm-generated signatures that are used in the queries. For this reason, the queries may use only the algorithms available in the server application.

In this extended abstract we briefly describe the client-server architecture of *NetPICSearch* (Section 2) and how the system is used to retrieve images over a network (Section 3). An extended version of this paper is to appear in [3]

## 2 Architecture of NetPICSearch

*NetPICSearch* is divided into a server and a client application. The server is a background process without a separate user interface. The server is used via the client application. The client application provides a graphical user interface for normal retrieval use as well as for the system administrators. The network protocol used in communication between the two applications is TCP/IP.

The client application is coded by using C++ and Tcl/Tk. The graphical user interface is implemented with Tcl/Tk to ease portability between different operating systems. Most of the actual functionality is implemented in C++. This includes the communication with the server application, file access and image manipulation.

The server application consists of several processes that are all programmed in C++. Client applications send connection requests for the server through a TCP port. A host process receives the requests and checks for adequate authorization before starting a dedicated server process. Further communication from the client application is handled by this dedicated server process. The host and all of the server processes use one common database process for database access. The relations between the main processes are illustrated in Figure 1.

A server process fulfills requests of a client application; it receives messages and interprets them. Before taking any actions, the server makes sure that the client has sufficient user rights to execute the requested operation. The most typical operation is an image retrieval query. The client provides a signature as a key for the query and the server utilizes the database process to produce a set of matched images. Finally the results are packed into a compact form and sent to the client application.

*The financial support of the Helsinki Graduate School of Computer Science and Engineering is appreciated.

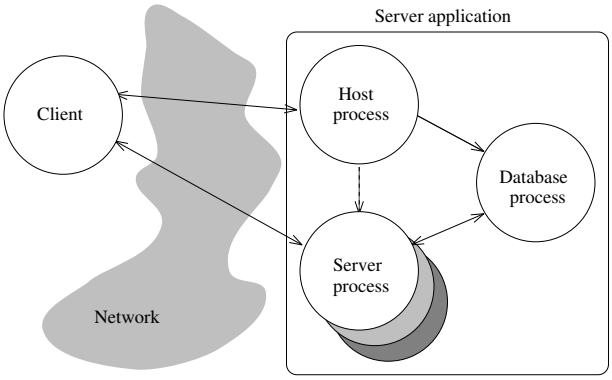


Figure 1: *The main processes of the server. The client can access the databases of the server through a server process.*

Both host and server processes need the database to operate. All these processes are served by a single database process. The database process communicates with the other processes through pipes and provides the basic database functionality. Server installations running *NetPICSearch* may offer multiple image databases for the client applications and each image database may contain a number of images and have user rights defined separately. All the image databases are still accessed through the same database process which hides the details of physical implementation.

# 3 Retrieving Images Over a Network

*NetPICSearch* can be used anywhere on the Internet (or within an intranet). Because the key functionality of *NetPICSearch* is located in the server program, a significant amount of network traffic between the client and the server is necessary. In a low bandwidth network this could slow down the system considerably. To avoid that, we have designed our communication protocol to use as short messages as possible.

Retrieval algorithms based on image content need the image as an input. However, transferring a large image over a network results in an undesirable delay. In *NetPICSearch* we normally do not need to send the whole image to the server. Instead, the client computes the signature out of the image. Signatures are usually much smaller than the whole image data and this makes the querying process more efficient because only the signature of the query image is sent over the network and traffic is thus minimal. Furthermore, when the server has executed the query, it sends only the thumbnails of the matching images to the client, not the actual full-sized versions of the images.

Since we use preprocessing of the database images and we want to minimize the messages (and therefore we do not want to sent an algorithm from client to server), only algorithms combined to the connected database of a server can be used. If we had enabled the sending of algorithms, the increased communication would not have been the only reason to slow down the retrieval, but also a remarkable time would have been spent on on-line computation of the signatures out of database images.

When the user has received the images provided by a query, (s)he may execute a query with one of the result images as the new query image. In this case the server does not need even the signature of the query image since the image already resides on the server. If the user has found an image that meets their demands, it can be transferred from the server just by clicking a button on the screen. After receiving the whole image, (s)he may connect to another server and start a new series of queries with this image. Thus, the querying process can be a long chain of queries where different servers and databases all over the Internet can be used to achieve a satisfactory result.

*NetPICSearch* provides also basic tools for the administration of the servers and their databases. Server administrators can create and delete image databases and specify different login names to give access to the images. Thus, we can create public databases as well as more or less restricted databases. Server administration, i.e., user management and database maintenance, may be done remotely (when the client and the server are running in separate machines) over the network via the graphical user interface of the client program.

# References

[1] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C. Shu. The Virage image search engine: An open framework for image management. In *Proceedings of the SPIE: Storage and Retrieval for Still Image and Video Databases IV*, February 1996.

[2] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, 1995.

[3] P. Kuusi, K. Lemström, P. Kanerva, S. Koponen, S. Männynsalo, J. Riihimäki, J. Teinilä, and T. Virtanen. A client-server tool for image retrieval applications. To appear in Proceedings of the Challenge of Image Retrieval (a Research Workshop on Image Retrieval), February 1998.

[4] K. Lemström, J. Korte, P. Kuusi, P. Kyheröinen, and P. Päiväkumpu. PICSearch - a platform for image content-based searching algorithms. To appear in Proceedings of the Sixth Int. Conf. in Central Europe on Computer Graphics and Visualization (WSCG'98), February 1998.

[5] S. Mehrotra, Y. Rui, M. Ortega, and T. S. Huang. Supporting content-based queries over images in MARS. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems ICMCS'97*, June 1997.

[6] S. C. Orphanoudakis, C. Chronaki, and S. Kostomanolakis. $I^2C$: A system for the indexing, storage, and retrieval of medical images by content. *Journal of medical information*, 19(2):109–122, March 1994.