Slovenian Virtual Gallery on the Internet

Peter Peer and Franc Solina
Computer Vision Laboratory
Faculty of Computer and Information Science
University of Ljubljana
Tržaška 25, SI-1001 Ljubljana, Slovenia
E-mail: {peter.peer, franc.solina}@fri.uni-lj.si

Abstract

The Slovenian Virtual Gallery (SVG) is a World Wide Web based multimedia collection of pictures, text, clickable-maps and video clips presenting Slovenian fine art from the gothic period up to the present days. Part of SVG is a virtual gallery space where pictures hang on the walls while another part is devoted to current exhibitions of selected Slovenian art galleries. The first version of this application was developed in the first half of 1995. It was based on a file system for storing all the data and custom developed software for search, automatic generation of HTML documents, scaling of pictures and remote management of the system. Due to the fast development of Web related tools a new version of SVG was developed in 1997 based on object-oriented relational database server technology. Both implementations are presented and compared in this article with issues related to the transion between the two versions. At the end, we will also discuss some extensions to SVG. We will present the GUI (Graphical User Interface) developed specially for presentation of current exhibitions over the Web which is based on GlobalView panoramic navigation extension to developed Internet Video Server (IVS). And since SVG operates with a lot of image data, we will confront with the problem of Image Content Retrieval.

1 Introduction

Think of a system that can be reached and managed from any part of the world, has a large text and image database, fast and efficient database engine, 3D explorer of data, is easy to maintain and add new functionality, provide you with life video information and has a good image query possibilities. We incorporated all of these ideas in a system called Slovenian Virtual Gallery (SVG). The system is a union of information systems techniques, database engine par excellence, UML (Unified Modeling Language) modeling, Internet techniques, Web design, GUI (Graphical User Interface) design, computer graphics and computer vision methods.

The goal of the Slovenian Virtual Gallery (SVG) is a multimedia presentation of Slovenian fine arts and architecture on the Internet [3, 4]. In cooperation with distinguished slovenian art-historians we prepared an overview of Slovenian art from the gothic period up to the present days. A selection of the greatest works of art, descriptions of the periods of art history, the artists' biographies as well as other information on the works of art and their authors is presented. Exhibitional activities of some major Slovenian art galleries are incorporated as well — included are the permanent collections and current exhibitions. SVG was warmly received in Slovenia as well as on the Internet in general. In 1996 SVG was rated by The McKinley Group's online editorial team as a "4-Star" site excelling in "Depth of Content", "Ease of Exploration" and "Net Appeal".

The World Wide Web [12] environment was the obvious choice for the implementation of the Slovenian Virtual Gallery. SVG employs all multimedia elements of the HTML language [13]; besides the standard interlinked combination of images and text also clickable images (imagemaps) and video. Access to information is in a typical hypertext fashion — clicking on parts of text or images leads to related or more indepth information. For example, clicking

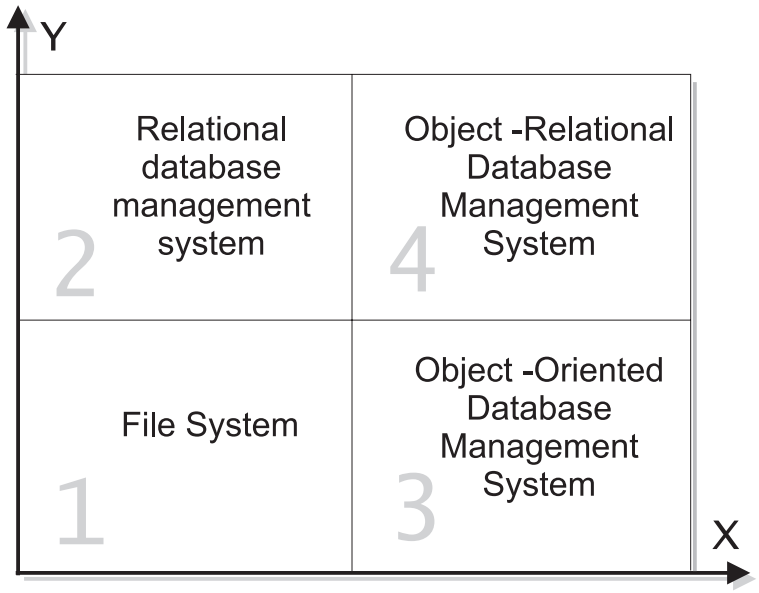

Figure 1: Data management technologies classes; the ability of managing complex data types is rising in x direction while in y direction the frequency of general data query is rising.

on the name of an author brings up his biography and a catalogue of his works included in the SVG.

In the time span of just a few years the Web related technology has experienced a tremendeous growth. For data management of web sites one can select from the following four classes of technologies (figure 1):

1. Data file systems
2. Relational database management systems
3. Object-oriented database management systems
4. Object-relational database management systems.

The first implementation of SVG was based on data file systems and custom software for data management and generation of HTML documents. The second implemententation uses a commercial object-relational database. In this paper we give a short description of both implementations and their pros and cons as well as issues of changing from the first to the second implementation of SVG.

The paper has the following structure. The second chapter gives an overview of the SVG contents and its hypertext features. In the third chapter the first data file based implementation of SVG is given. The fourth chapter gives a description of the Universal Server, which was used for the second implementation of SVG. We will list the problems regarding transfer of an existing web site to a newer technology. At the end of this chapter, we also compare both techniques. In chapter five we will discuss our approach to effective presentation of current exhibitions over the Web, introducing our GlobalView extension to developed Internet Video Server (IVS). The sixth chapter introduces some results of image based content retrieval methods implemented in our laboratory. In the last chapter, you will find some directives for future work.

2 The contents of the Slovenian Virtual Gallery

There are three main parts of the SVG (figure 2) visible to a user:

1. **Presentation of the periods of Slovenian art history.** This is a somewhat classical hypertext structure linking pictures and text. For each art history period a brief description, a list of authors and a iconized index of works is given. The information on individual authors consists of a short biography and an iconized list of his works which are included in SVG. Each icon can be blown up to the screen size. Individual works

Figure 2: Front page of SVG.

of art are identified by its author, year of creation, technique, dimensions, signature, location and possible additional descriptions. Links allow a non-linear exploration of the text and images. See figure 3.

2. **Permanent collection and current exhibitions in Slovenian art galleries.** The galleries that participate in SVG are presented through a multimedia description of their history and activities, including text, images and video. Permanent collections and current exhibitions are included. When a new exhibition is opened, the previous is shifted onto the list of old ones and remains within reach of the SVG system. See figure 4.
3. **3D virtual gallery.** This is a virtual exhibition place. We created an interactive model of a threedimensional architecture as a virtual gallery space. It is possible to move inside this space by clicking on the part of the space where one would like to go (or on the little navigation balls) which brings the observer to the next view location. By clicking the pictures which hang on the walls the corresponding picture is shown upfront where links to the author and other parts of SVG are provided. See figure 5.

Among other things, SVG enables users to search for desired authors or works of art by different keys: by name, title style, the year of creation, location, etc. So the user can find for example all Johns - expressionists, or for example all the abstract paintings painted later than 1950 in oil on canvas.

Figure 3: Author biography.

3 File based implementation of SVG

From the very beginnings of SVG in 1995 we strived for a robust, adaptable but efficient system [2, 6]. Due to the lack of appropriate tools we decided to use plain text files as a database and wrote our own data management tools and tools for generating HTML document. Images were stored separately from other data in their own natural formats (JPEG and GIF).

For implementation we used PERL[14] because it enables simple manipulation of files and it similarity to C.

Because of the huge amount of data and because of the simpler maintenance the whole project is supported with a database management system which makes the inevitable work of the manager of SVG much easier.

The main advantage and reason for the introduction of database is the automatic generating of HTML documents. High typicality of some groups of documents (the presentation of an author or exhibition, the current exhibition of the gallery, the main exhibition room, various indexes of authors and works of art etc.) allows us namely to generate them automatically using adequate patterns. So for each work of art the backbone for the database record is automatically written and smaller versions of pictures are created (in the system SVG three versions of the same picture exist: the icon, the presentation picture and the picture of

Figure 4: Gallery description.

the final resolution). When additional data is entered to the database, it gets designed on a base of the adequate pattern and with other elements (pictures, text, links) integrated into HTML document. The system administrator will not need to know the syntax of HTML language, server's operating system nor will he need to process the pictures. All the above mentioned activities will happen with a click on the appropriate option of the program which combines all these functions and has been called the SVG Remote Manager.

3.1 Distributed database

Another criteria when designing the system was openness. By the rapid development of the Web and the dynamic quality of the media a statically designed system could not survive. This is the reason why the database was constructed distributively; parts of SVG can be located in various physically distant places. It follows that for example a gallery (or a period of art history) can simply be moved on another computer. A Slovenian gallery from abroad could be included as well, which clearly reflects the transteritorial nature of Internet.

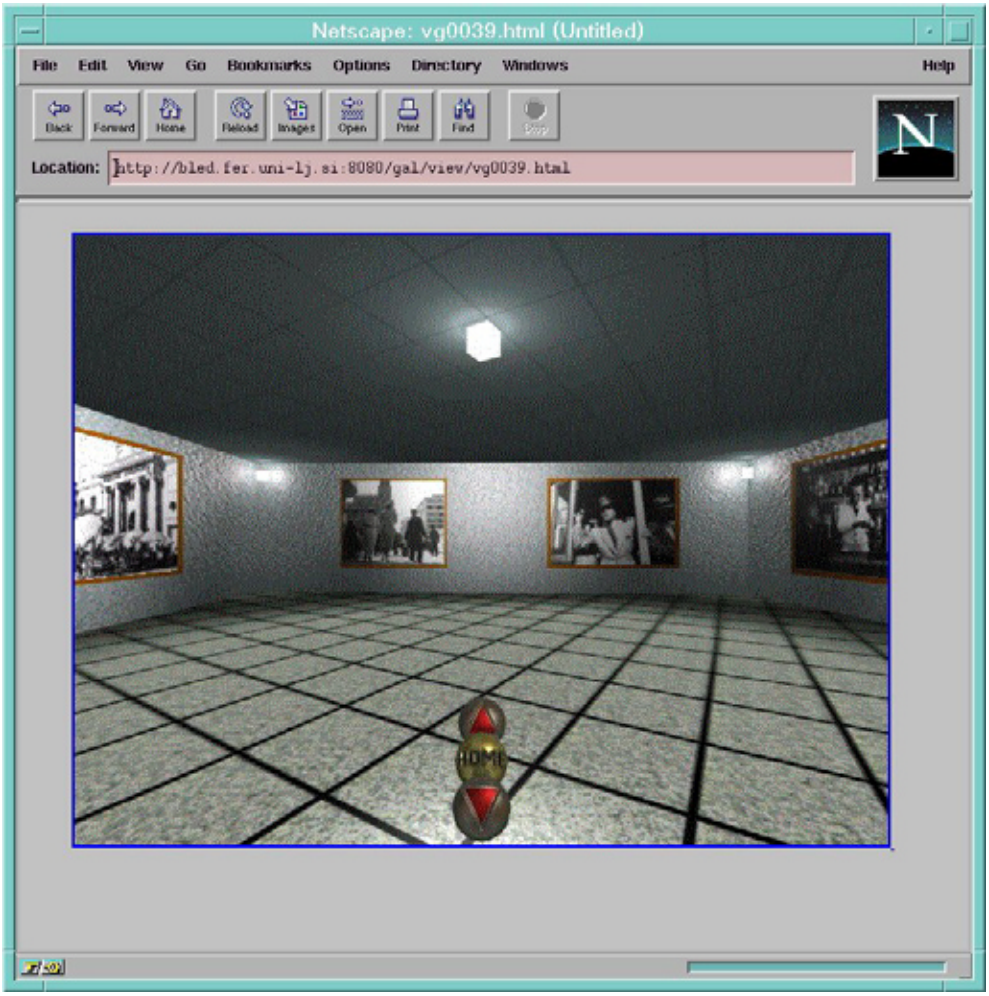

Figure 5: 3D virtual gallery space.

3.2 Generation of HTML documents

Most HTML documents are generated automatically from predefined forms for each SVG element (presentation of individual artists, current exhibition in a gallery, lists etc.).

SVG_DB module can extract the data from database file and return the data like rows of data file in any format or like macros for M4 [16] or HTMLPP [15]. M4 is a standard UNIX macroprocessor while HTMLPP is HTML specific macroprocessor written in PERL. SVG_DB can be used in UNIX pipeline and is independent of database format.

3.3 Search

SVG_SEARCH module enables end-user to search the SVG pages. It works in combination with HTML user interface, which offers interaction with end-user through forms. Module executes on web client demand when end-user - visitor fills the form and launches the query.

3.4 Remote management

The Slovenian Virtual Gallery is more than just a collection of static HTML documents. We have developed a tool, which makes possible remote managing (figure 6) of the system over the Internet and therefore enables an effective and at the same time simple managing of the data to the administrator of SVG. An authorized administrator can manage the data from a distant location and in a simple way generate final HTML documents that Slovenian Virtual Gallery consists of.

Therefore SVG Remote Manager is a tool enabling simple managing of the whole system remotely over Internet. It is a universal program, which makes it possible for the authorized

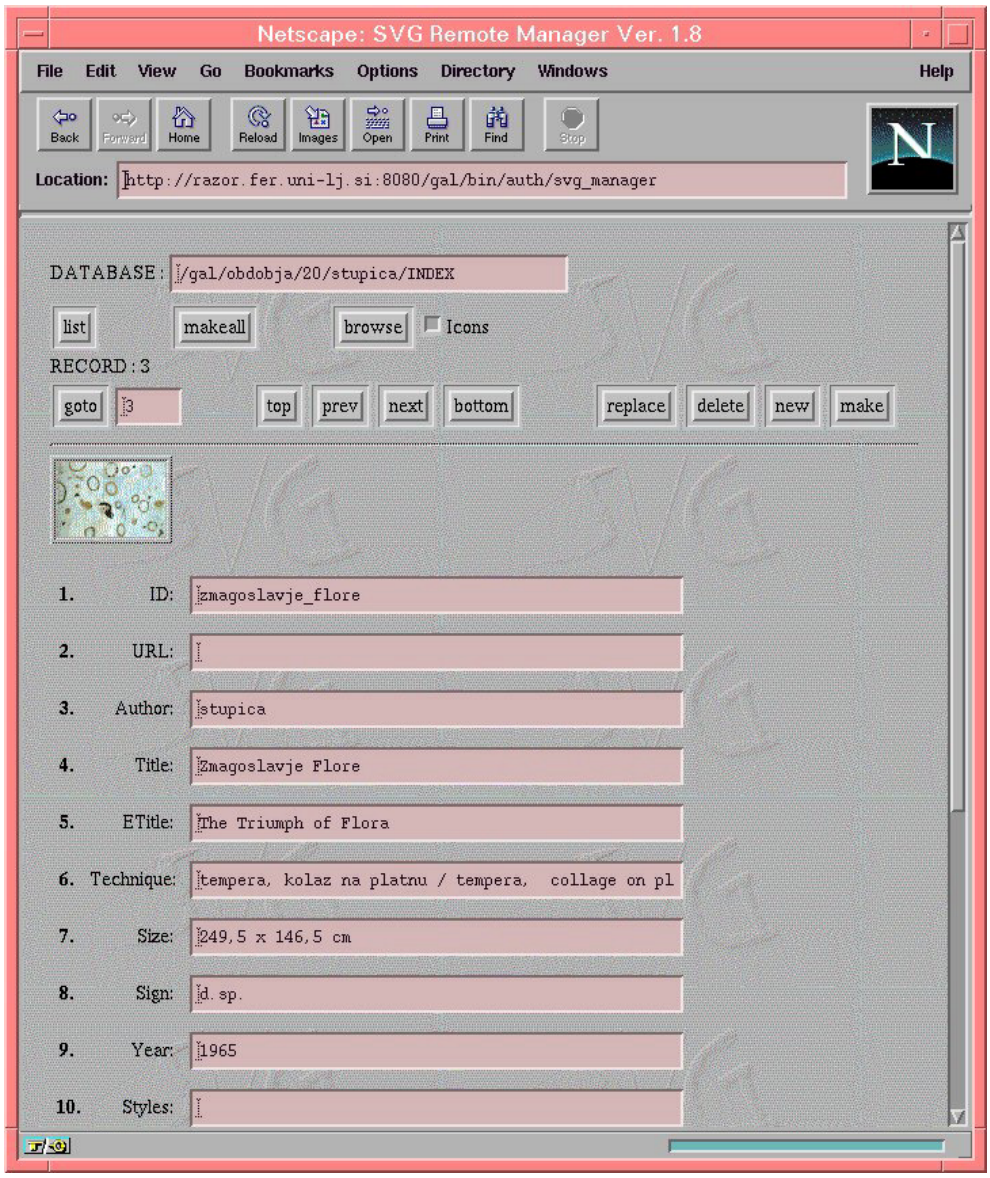

Figure 6: Management window.

administrator (there can be more administrators, each can handle a specific area, which guarantees more competent data) to work from a remote terminal on any platform.

3.5 SVG as a virtual exhibition space

A major part of our picture collection is also accessible through a 3D virtual gallery space. This is a virtual space in a form of a gallery, which was constructed by using 3D computer graphics. The idea was to enable the viewer the experience of visiting a gallery and to see the paintings in a more natural way.

To simulate the actual experience of seeing an exhibition in a real-world gallery, we had to enable a walkthrough the gallery rooms and the observation of the paintings hanging on the walls. At this moment there are two ways of achieving this goal. The first is by the use of VRML. The other way is achieved by the use of the image-map mechanism (which is incorporated in the HTML specification) in a special way; each picture is a pre-rendered view of our 3D world and incorporates carefully selected links to the next view. By clicking on a desired area of the picture, we move to the correspondent destination. Thus the selected sequence of image-maps forms a walk through the gallery. This approach has also some other advantages over the VRML model:

- Every view in a sequence is a pre-rendered picture (a simple GIF or JPEG file), which can be limitlessly complicated - it could even be a photograph of a real situation.
- Every walk step is an HTML document. This provides an enormous flexibility and all the connectivity we want. In our case we used it to link the virtual 3D space with the contents of other parts of SVG.

The 3D virtual gallery is currently implemented as a structure of XY inter-connected clickable-maps. Clickable-maps are pre-rendered views of the virtual gallery that was constructed as a classical CAD model and rendered with *3D Studio.* The topology of the space is governed by two main factors: the number of galleries included in SVG and the number of the works of art in a gallery. The result is a symmetrical structure and each part belongs to one gallery. The corridors of the galleries are of variable length.

As mentioned before, the 3D virtual gallery is not just an isolated virtual world. It is closely integrated with other parts of SVG. Namely, there is a passage through every work of art in the virtual gallery — by clicking on a painting hanging on the wall, we arrive at the presentation of the painting along with all the correspondent information. From the presentation of the painting, we can then move forward to the presentation of its author and from there to other works from his opus.

3.6 Drawbacks of a data file based implementation

The solution for storing data in the first implementation of SVG is effective and cheap but also has some drawbacks:

- Integrity and safety of the data is not provided.
- Data model is also encoded in software, although not completely.
- Appending and content changes demand complex corrections.
- Software maintenance is problematic.
- Changes in functionality lead us to non-optimal processing.
- In a case of large database the system is not robust enough and is too slow.

Having this in mind, we implemented the newer version of SVG as an object-relational database management system.

4 Universal Server

The second version of SVG [5] dispatches problems mentioned in previous section and ensures data queries based on Standard Query Language (SQL). With this, we get flexibility of answering the questions that are not known in advance. Object- oriented system gives us natural assistance in using complex data types. From the user's point of view multimedia objects are stored in the same way as any other data and at the same time the majority of program logic is stored in server which dispatches the need for set of additional specialized programs.

Universal servers are object-relational database systems, open- designed, offering expandable user defined data types and enabling users with power of using specialized complex types together with operations on those types. We implemented the second version on Informix Universal Server. The fact that system is open-designed is very important since it delivers easy upgrading with new data structures and operations. This property is called DataBlade (data module) and adds new functionality to universal server.

4.1 Informix Universal Server

Universal server introduces DataBlade Application Programming Interface (API) for development of DataBlades. Therefore, it is possible for independent software developers to provide highly specialized object modules with full functional support for object management.

Informix Universal Server (IUS) is a very efficient data server with high parallel level in query solving. It executes dynamic means evaluation for optimal task classification and supports Symmetric MultiProcessing (SMP) computer systems, where the operating system is executed on parallel central processing units which all share common memory space and

communicate with each other. Data tables can be divided in fragments, which enables us to perform parallel processing of user-defined queries. IUS uses direct access to secondary memory modules (discs). That means it dispatches constraints of file system, optimizes the speed of accessing data tables and facilitates the transfers between primary and secondary memory modules. It manages dynamically with common memory space and uses different methods for safety and reliability of data storage.

Figure 7 illustrates IUS architecture. DataBlades access the server through common program interface and are tightly bound in server's kernel. Functions consisting DataBlades use common memory space, directly use server's complex data types and are at the same time object of query processor optimization.

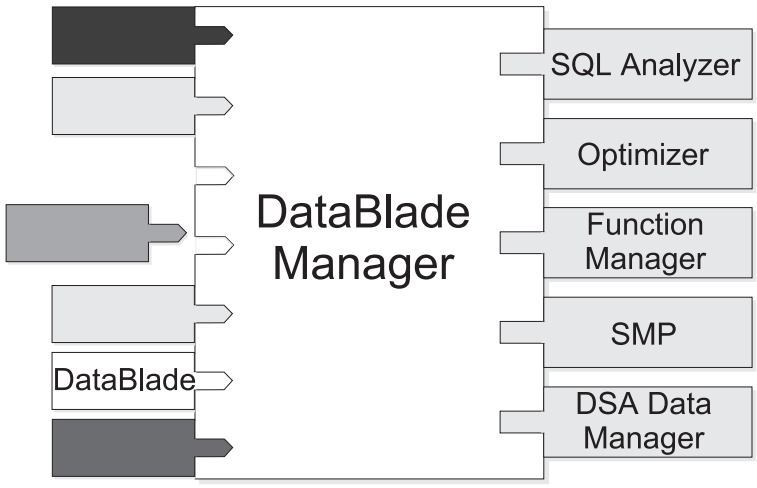

Figure 7: Informix Universal Server architecture.

4.2 DataBlades

DataBlade is a set of objects and algorithms, which extend the universal server functionality. New objects become a part of server and are equivalent to already existing data types. This means we manage them through SQL statements.

Domain spectrum for development of DataBlades is practically unconstrained while module can implement any functionality. Let's mention some developed DataBlades:

- Image DataBlade - image content search and image processing
- LOBLocator DataBlade - working with binary objects
- Web DataBlade - web application developing module
- Video DataBlade - working with video clips
- Text DataBlade - text processing
- DesCrypt DataBlade - cryptography module
- Geodetic DataBlade
- Face Recognition DataBlade
- Audio Information Retrieval DataBlade

DataBlade consist of user defined data types and functions written in C or SPL (Stored Procedure Language). We would expect support for C++ since we are dealing with object-relational database management system but unfortunately, this is one of the major drawbacks of IUS. Functions are then executed in server's kernel.

SVG uses Web DataBlade, Image DataBlade, LOBLocator DataBlade and some user-defined DataBlades.

4.3 Application Pages and Dynamic Tags

Web applications dynamically generate web documents. They use Common Gateway Interface (CGI) or its extension like NSAPI or ISAPI and enable interactivity, generation of web documents on web client demand. Interaction with user can be achieved through HTML forms or applets, which are applications written in Java, and executed on web client.

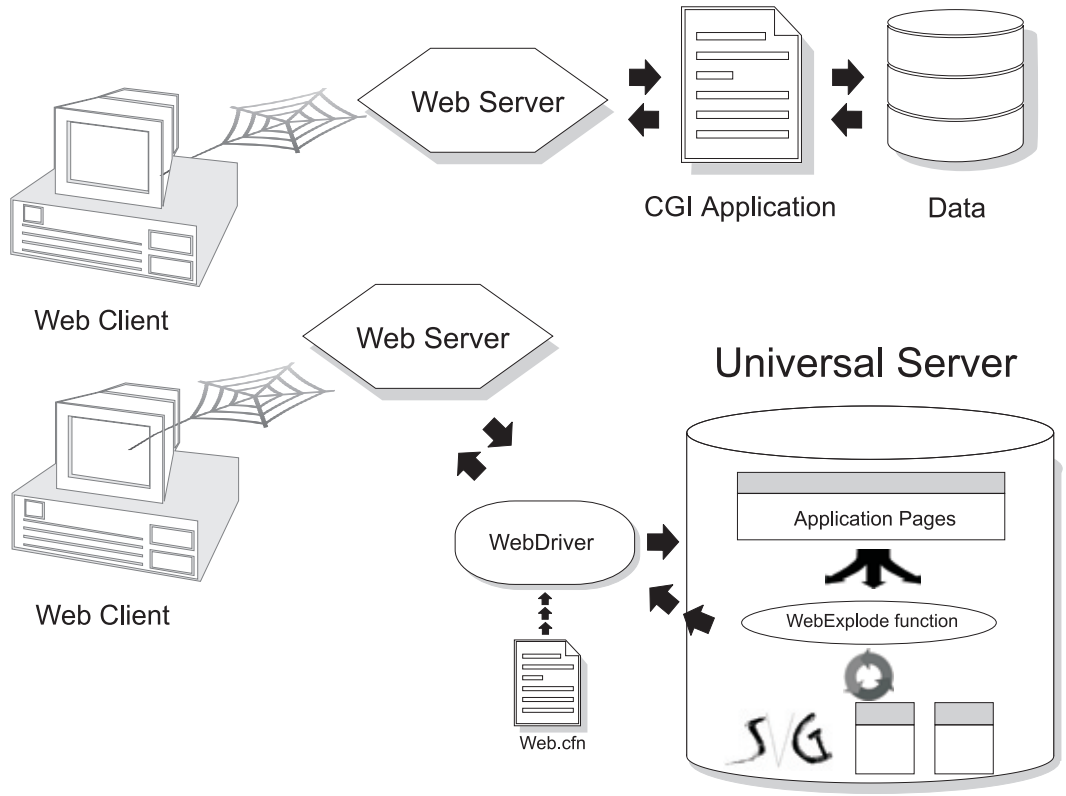

Figure 8: Difference between standard web page serving through CGI and architecture of Web DataBlade.

Figure 8 illustrates the difference between standard web page serving through CGI and architecture of Web DataBlade. IUS still uses CGI (or NSAPI) application WebDriver to connect IUS with web server but here WebDriver serves only for delivering client demands to data server and does not contain the logic of SVG. In the second case, SVG logic is stored inside database in the spirit of object-oriented technology, like set of web page objects. Objects of that type are in the terminology of Web DataBlade named Application Pages.

Application Pages are web pages written in HTML and extended with additional elements called Dynamic Tags. Dynamic Tags follow the Standard Generalized Markup Language (SGML) specification and contain SQL statements, which are processed by universal server. Universal server performs the query in the database and returns results in proper format. Therefore, Application Pages are templates for construction of web pages while Dynamic Tags implement the functionality of CGI applications used in the first version of SVG.

We can divide Application Pages program logic in three groups:

- Application Pages with Dynamic Tags
- Functions within server: procedures stored in database and functions within DataBlades
- Applets.

First two groups are executed on server side while the third group is executed on client side. SVG has parts in each group.

4.4 SVG on Universal Server

Transition of first SVG system to second version based on universal server technology demanded transport of all the data and program logic. We transported the data from plain files to database tables and rewrote all the program logic since now it is mostly hidden within Application Pages in the form of SQL statements and partly implemented in IUS DataBlades and applets.

Data transition process demanded creation of new database, combining data from non-standard form of files, renaming values of primary key identification columns, generation of links and data filling. Work was mainly done automatically with help of shell scripts under UNIX.

Because of compatibility with first version of SVG we developed administration tool SVG Remote Manager v2.0. It has some improvements but basically performs the same functions as in the first version, although it's not so important in the second version of SVG as it was in the first version of SVG due to other possibilities of data management. The second version of SVG Remote Manager adds possibility of sending files from web client to web server, which enables us to add image and text data directly into database from local computer and changing the SVG Application Pages. Now we are finally able to manage SVG completely from a remote computer. One of the main features of SVG Remote Manager are still simple graphical user interface for data input and online data control which prevents the possibility of administration errors.

We also took advantage of one IUS feature called Image DataBlade. DataBlade was written by Excalibur company and works on the principle of shape, color and structure similarities and provide us with some basic image processing algorithms. In SVG system one can search for images according to:

- similarities to the art presented on the web page
- similarities to the uploaded sample image
- Java application SVG Image Finder where the user can draw the sketch.

4.5 Comparisons

The main advantages of new implementation are:

- robust and standard database management
- simple software maintenance
- improvements and new tools
- system openness, simple changing and functionality upgrading.

Figure 8 gives the difference of serving web pages in first and second version of SVG. In the first implementation web server serves beforehand prepared documents, while in the second implementation documents are build for every client access. In the first version the data is doubled, i.e. data stored in database and static web documents, enabling fast demand serving and the second version waste the processor power, but since IUS supports SMP systems and cached mechanisms the biggest problem remains the internet connection between client and server.

Application Pages are a very good idea but have some drawbacks:

- they are not entirely undependable from data model
- problem of nesting Dynamic Tags
- some awkwards while working with variables.

5 Effective Presentation of Current Exhibitions over the Web

Life video transmission over the Internet and interactivity are becoming more and more popular. At this moment hundreds of cameras, all across the world can found on the Web that can be used as remote eyes. Video can provide information that static images can not (telepresence) and with further development of technology and Internet infrastructure the

speed of transmission and the amount of video imagery will only increase. Therefore, intelligent control of video capture by means of changing the view direction of the camera, spatial structuring of visual information, as well as generation of visual summaries are essential for successful application of this technology. To study user-interface issues of remotely operable cameras and provide a testbed for the application of computer vision techniques (motion detection, tracking, security) the Internet Video Server (IVS) system was developed which was recently expanded with the GlobalView interface [8, 9].

5.1 Internet Video Server

IVS enables live video transmission and remote camera control (pan and tilt) over the Web. In designing the system, certain objectives were followed:

- client side platform independence
- optimization for slow connections
- remote control of the camera.

Platform independence of client was easily achieved by using the Web technology. On almost any platform, one can find a Web browser capable of displaying text and graphics.

To achieve greater flexibility of operation, the camera can be placed at a location with or without a slow Internet connection. This is made possible by a two-level IVS system architecture. As shown in figure 9, the first level is the distribution level and the second is the camera level.

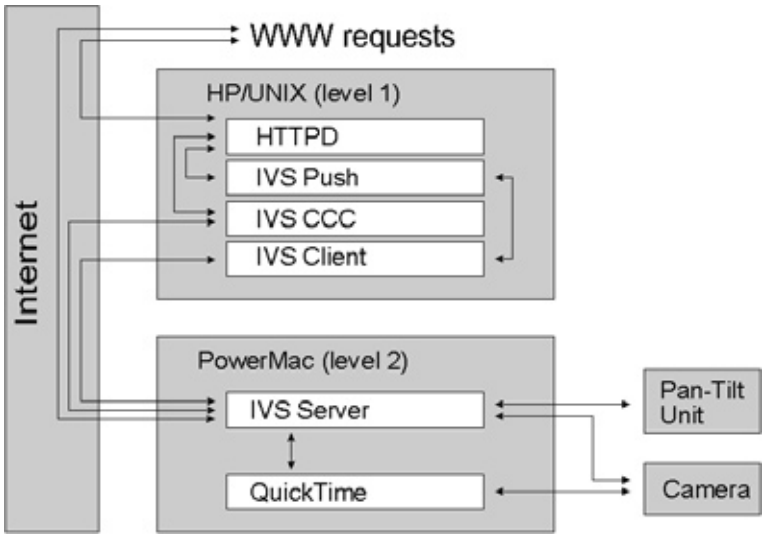

Figure 9: IVS architecture consists of two levels: distribution (1) and camera level (2).

5.2 GlobalView Extension

It was observed that the user of IVS had some problems operating the camera with the interface shown in figure 10, which become more severe when the response of the system was slow due to slow connections. A user would, for example, press the button for moving the camera one step to the left. If nothing happened in a few seconds, he or she would press again the same button or try a different operation. Due to buffering, he would suddenly get a new image, which could be a result of just the first command, or a combination of any number of subsequent commands. Due to slow and uneven reaction of the IVS system to user's actions, the operation does not seem to be very predictable from the user's viewpoint.

Another problem with the user interface is more perceptual. If the focal length of the lens is large, one can easily lose the notion to which part of the scene the camera is pointing at. Observing a distant location through IVS or a similar system gives a somewhat limited perception, akin to looking at the surrounding through a long tube.

Due to the precisely controlled position of the camera by means of pan-tilt unit, individual images acquired by IVS can be assembled in a panoramic view of the entire surroundings

Figure 10: The plain IVS user interface showing buttons for controlling the camera directions.

which can be used as a backdrop for the current live image! If we want to find some object on the scene and for that we need to preview, the whole scene at least 100 camera movements are required. Using an updateable panoramic view as a part of the user interface the number of camera movements required to find some target can be decreased significantly: to one move only!

The IVS system was expanded with the GlobalView interface to give IVS users a better spatial perception of the observed location and more intuitive control of the camera platform. We implemented two methods for static panoramic image generation: one uses geometric transformation and the other is the brute-force scanning approach [8]. GlobalView interface (figure 11) is a combination of a static panoramic view and live images received from IVS. Live images arriving from IVS are transformed from spherical to cylinder coordinates and superimposed on the corresponding position in the panoramic view.

At the system startup, the panoramic image is first generated by scanning the complete surroundings of the camera. When a user starts interacting with the system he or she is shown the whole panoramic image. A rectangular frame in the panoramic image indicates the current direction of the camera. Inside this attention window, the life video image is seamlessly superimposed onto the panoramic image. By means of mouse the user can move the attention window and in this way select the direction of the camera. When the attention window is at the desired location the user prompts this to the system. From the position of the attention window within the panoramic image the physical coordinates of the next camera direction are computed and the appropriate command is issued to the pan-tilt unit. When the camera is moved to the new direction, the last image from the old position is pasted to the static panoramic image. In this way, the panoramic image is constantly updated in the areas of most interest to observers.

Figure 11: GlobalView interface of IVS. The rectangle in the panoramic image indicates the attention window, which in turn indicates the current direction of the camera. By moving the rectangle, the user controls the direction of the camera. On the bottom of the web page the live video image in actual resolution (left) and a zoomed panoramic view (right) are shown.

The client-side user interface was written in Java (Java applet) which is run within any web browser that supports Java. This approach enables platform independence. With minor changes, it can be run on any Java virtual machine.

5.3 GlobalView and IUS

When using the IVS it sometimes needs to be seen what was happening at a remote location in the past. For example, a user might want to check who entered a building in some specific time interval or just to keep track of the number of people in the specific location.

For the generation of visual summary reports the Informix Universal Server is being used, which enables image base queries and reports.

6 SVG Image Content Retrieval Methods

Since SVG operates with a lot of image data, we were confronted with the problem of Image Content Retrieval. As mentioned we took advantage of one IUS feature called Image DataBlade. DataBlade was written by Excalibur company and works on the principle of shape, color and structure similarities and provide us with some basic image processing algorithms. However, since we are computer vision people, we were interested in the background of these approaches. In this chapter we will introduce some results of image based content retrieval methods implemented in our laboratory [1, 10, 11].

6.1 Using Color Features

Our goal of this project was to adopt the general machine learning methods for the use in image retrieval systems [1]. We examined three different machine learning methods (k-nearest neighbors, ID3 and Naive Bayes classifier) and made small adaptations to incorporate them in a web based search engine.

Machine learning methods can use different types of attributes like color attributes, texture and shape. The most popular are color attributes as they can be computed fast and in a straightforward manner. Several types of color attributes and their combinations can be used (histograms, moments, primary colors, averages etc.) which can also be computed separately in predefined parts of the image. Color attributes are not sensitive to location, rotation, scale and resolution. On the average, they give good results but they miss images, which are to a human observer very similar but of different colors.

Let's list color attributes we are using. - Color histograms are of high dimension and therefore it is difficult to compute the distance between them. The first and second moment of color histograms are much more compact and easier to compare. The **first and second moment of color histogram** is the average RGB color and its dispersion. In our system, they are computed on the whole image and in the central part (middle three fifths) of the image. **Compactness of color** measures the proportions of pixels of "mostly red", "mostly green", "mostly blue" and "gray" colors surrounded by pixels of similar color. The pixel of color (r, g, b) is "mostly red" if $r - \max(g, b) > \delta$, "mostly green" and "mostly blue" are defined similarly, all remaining pixels are "grey". **Proportions of basic colors** are proportions of pixels of "mostly red", "mostly green", "mostly blue" and "other" colors.

The color attributes are precalculated for all images in the database. The computation of all color attributes for 1000 images takes less than 10 minutes on a PC. This set of attributes serves as input to the learning part of the system.

First the system presents the user a certain number (15) of randomly selected images and asks him or her to grade them (figure 12). Each image can be given one of five grades, with the lowest meaning that the image is completely different from what he or she looks for and the highest meaning that the image is of exactly the right type. The user is not required to classify all the images.

The data posted, grades are converted to classes and weights. The images having the middle grade are skipped. The lower two grades are converted to "NO" class with the lowest having weight 1 and the other 0.5. The higher two grades correspond to "YES" class with the highest having weight 1 and the other 0.5. The precalculated attributes together with the just constructed class can weight values are given to the learning algorithm. The obtained classifier is used to estimate the probabilities of "YES" class for all other images in the database which have not been presented to the user yet. The fifteen images with the highest probability of "YES" class are presented to the user as an answer to his or her query and examples for its refinement.

If the query was unsuccessful, the user can grade the presented set of images to point at the good and bad examples again. The images are added to the previous examples. The learning algorithm re-learns with the new examples and a new selection of fifteen images is presented to the user again. In the case of a satisfactory answer, the user can refine the query by being more selective when assigning the good grades. The system decreases the weights of images of past queries and thus gives the new examples a bigger importance. The user can count on that and request a larger concept in the beginning and narrow it (become more strict) later, without the good grades from the first rounds of the process interfering in the later rounds.

Finally, if the query was a complete success, the user can request the next fifteen closest images using exactly the same classifier as before.

Experiments show that the most promising method for now is k- nearest neighbors, especially for its ability to work with smaller example sets than the smarter methods. This is not surprising; the fact is that most of working systems for image retrieval already use a simplified version of this method. Its performance could be further improved by refining probability estimation function and how it is influenced by the learning examples of different classes at different distances from the example, which is being classified. K-nearest neighbors' method proved to be efficient in retrieving images of faces, animals, landscapes, cityscapes and similar. It is also fast enough although the other two methods were a bit faster.

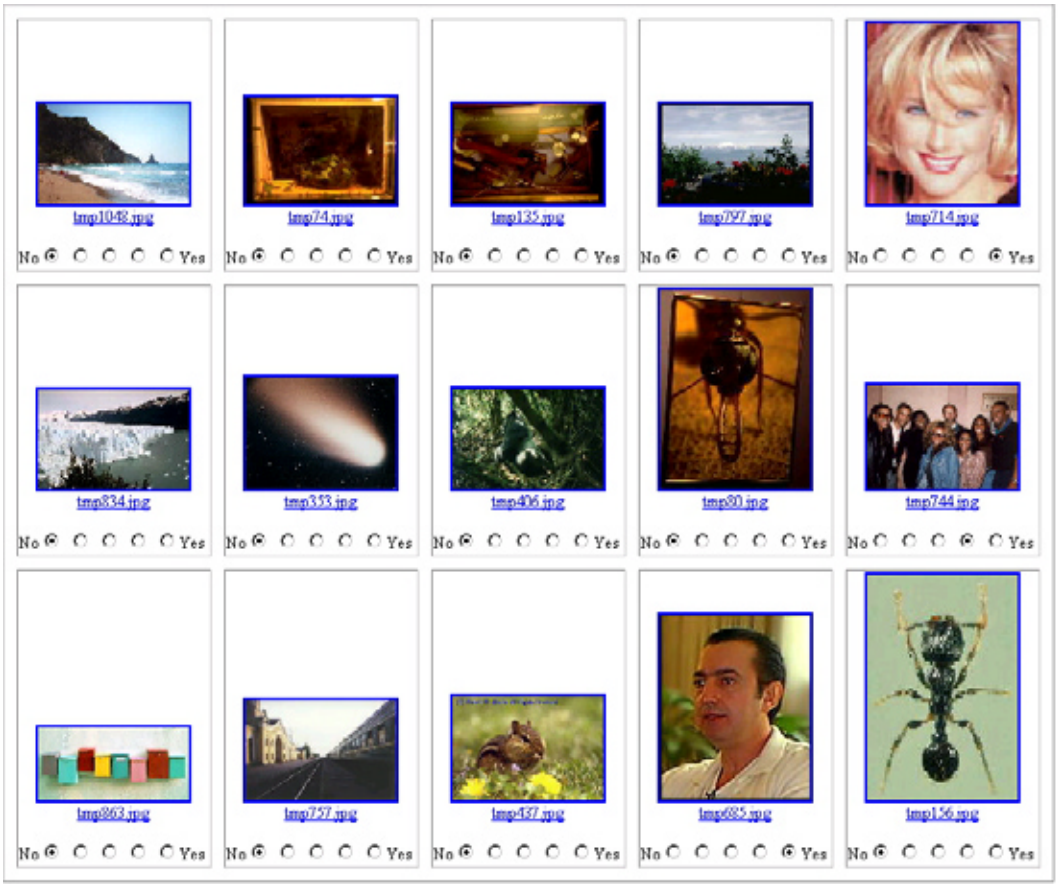

Figure 12: An example of a query for human faces. The user gave the highest grade to the two faces and the second highest to the image of a group of people. All other images have the lowest grade.

6.2 Using Interest Points

Shape is much more difficult to define and compute than color attributes. Even if one can reliably recover shape attributes the definition of similarity or the distance among different shapes poses a very difficult problem. Nevertheless, since the human perception of similarity of images can not be reduced only to color and texture, this area of research is very important. Due to computational complexity, the current shape attributes used in the framework of image retrieval are limited mostly only to edges, corners and interest points. For shape attributes it is particularly more difficult to attain location, rotation and scale invariance.

We examined the possibilities of a method that extracts similar images independently of their color information using interest points to mark image features or contours (i.e. edges, corners) [11]. Harris and Stephens combined corner and edge detector was used to determine interest points in order to identify images in a large database using a rotated, scaled or partly visible input image. The main question was whether the repeatable results of Harris and Stephens's detector are useful in retrieving similar images as well. This detector was used to determine candidates for interest points and then thresholding and reduction were used to obtain a smaller group of points that would give as much information about an image as possible. The positions of these points were transformed into rounded and normalized polar coordinates, giving a universal representation for each image. Set of such coordinates was used as image features and the distance function between two images was based on number of interest points with the same position in both images. Figure 13 explains the system behavior.

We proved that Harris and Stephens corner and edge detector is a very useful for querying similar images by interest points matching. Using interest points' location in polar coordinates also proved suitable. Main effort in further development should be put in modifying the distance function.

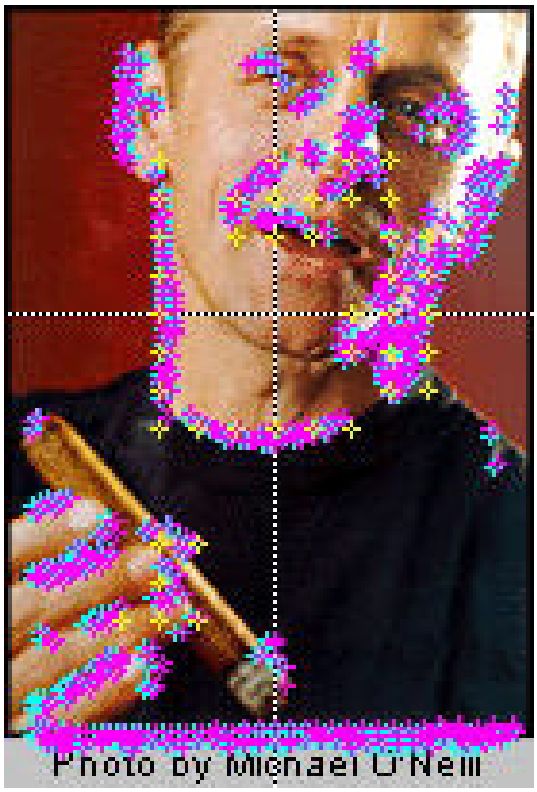

Figure 13: A sample of an output image, created by the image feature calculation program. Small crosses of different colors indicate candidates for interest point, actual interest points in polar coordinates and "heavy" points that help determine the "logical" center. The center itself is shown with a dotted cross through the whole image. Such an output image helps the designer while setting the parameters as well as assists in result evaluation since it helps to explain system behavior.

Naturally, the method was not expected to outperform the methods based on color information. However it was able to retrieve similar images to some extent and has fully met the expectations as color independence is considered.

7 Conclusions and Future Work

Slovenian Virtual Gallery project is an example of using new technologies par excellence while it evidently shows the advantages of using internet and interdisciplinary approach. SVG is not a commercial project; it's more of a pilot project illustrating use of new technologies, which will serve us in a future like a testbed for technologies to come.

We are developing a program that will automatically construct a complete VRML model of the 3D gallery. This will enable the 3D gallery to reflect the constant changes: the arrival of new works of art and inclusion of new galleries. The VRML version of the gallery will be available as an additional option at the front page of Slovenian Virtual Gallery.

Future work regarding Internet Video Server is directed to integration of visual information from several cameras (i.e. relay-tracking) and visually-based teleoperation of a mobile platform.

It is obvious that the given color attributes described in section 6.1 do not describe images precisely enough. Therefore, the future work shall focus mostly on searching for new image describing features.

With further development of interest point matching method (section 6.2), it could be useful as a foundation for a contour- based image retrieval application. Even more prospects can be seen if used in combination with other developed methods.

Acknowledgement

We thank Andrej Lapajne, Bor Prihavec, Aleksandar Ruben, Žiga Kranjec, Janez Demšar, Dragan Radolović and Stanislav Rozman who participated in the different stages of the SVG project. We also thank the art historians dr. Samo Štefanac, dr. Tomislav Vignjevič, Matej Klemenčič, dr. Barbara Jaki and dr. Igor Zabel for selecting the exhibits and the accompanying texts. Cooperation of Slovenian National Gallery, Museum of Modern Art, Mestna galerija, as well as galleries Insula and Equrna in Ljubljana is also gratefully acknowledged.

Some parts of the SVG project were supported by Ministry of Science and Technology of Republic of Slovenia (Project L2-8721) and the European Cultural Month Ljubljana 1997 (Project Netropolis—Cyborg's Eye).

References

[1] J. Demšar, D. Rodolović, F. Solina, Image Retrieval System Based on Machine Learning and Using Color Features, *Proceedings of the 8th International Conference on Computer Analysis of Images and Patterns*, Ljubljana, Slovenia, pp. 480-488, Springer, 1999.

[2] A. Lapajne, B. Prihavec, A. Ruben, Ž. Kranjec, *Slovenian Virtual Gallery* (in Slovene), Faculty of Computer and Information Science, University of Ljubljana, 1995.

[3] A. Lapajne, Slovenian Virtual Gallery (in Slovene), *M'ARS, Magazine of the Museum of Modern Art Ljubljana*, VIII(3-4), pp. 87-92, 1996.

[4] A. Lapajne, Art on the Internet, *Moj mikro* (in Slovene), No. 1, 1996.

[5] A. Lapajne, *Slovenian Virtual Gallery on the Internet — from a file system towards an object-oriented relational data base system*, B.Sc. Thesis (in Slovene), Faculty of Computer and Information Science, University of Ljubljana, 1997.

[6] P.Peer, A. Bežek, *Re-Engineering Slovenian Virtual Gallery* (in Slovene), Faculty of Computer and Information Science, University of Ljubljana, 1997.

[7] P.Peer, *Re-Engineering Slovenian Virtual Gallery with Unified Modeling Language* (in Slovene), Faculty of Computer and Information Science, University of Ljubljana, 1999.

[8] B. Prihavec, F. Solina, User interface for video observation over the internet, *Journal of Network and Computer Applications*, 21, pp. 219-237, Academic Press, 1998.

[9] B. Prihavec, *A system for live video transmission and active observation over the Internet*, M.Sc. Thesis (in Slovene), Faculty of Computer and Information Science, University of Ljubljana, 1999.

[10] D. Radolović, *Image database queries based on color information*, B.Sc. Thesis (in Slovene), Faculty of Computer and Information Science, University of Ljubljana, 1998.

[11] S. Rozman, F. Solina, Image database queries based on interest points, *Elmar VIProm-Com '99*, pp. 263-271, Zadar, Croatia.

[12] The World Wide Web Initiative: The Project,
`http://www.w3.org`

[13] HyperText Markup Language: Working and Background Materials,
`http://www.w3.org/hypertext/WWW/MarkUp/MarkUp.html`

[14] PERL — Practical Extraction and Report Language,
`http://www.ijs.si/perl/`

[15] HTMLPP, un preformateur de html?,
`http://acacia.ens.fr:8080/home/nthiery/htmlpp/`

[16] m4 — macro processor, *HP-UX manual pages.*