

PRISMA-MAR: AN ARCHITECTURE MODEL FOR DATA VISUALIZATION IN AUGMENTED REALITY MOBILE DEVICES

¹Mauro Alexandre Folha Gomes Costa, ¹Bianchi Serique Meiguins, ²Nikolas S. Carneiro
and ^{3,4}Aruanda Simões Gonçalves Meiguins

¹*Programa de Pós-Graduação em Ciência da Computação – Universidade Federal do Pará*

²*Faculdade de Computação – Universidade Federal do Pará*

³*Programa de Pós-Graduação em Engenharia Elétrica – Universidade Federal do Pará*

⁴*Rede de Informática LTDA - IT Research Manager*

ABSTRACT

This paper proposes an extension to mobile augmented reality (MAR) environments – the addition of data charts to the more usual text, image and video components. To this purpose, we have designed a client-server architecture including the main necessary modules and services to provide an Information Visualization MAR experience. The server side includes face recognition using OpenCV, and a visualization engine, using the PRISMA tool. The client side is focused on presentation and interaction, with a prototype developed for the iOS6 operational system and compatible with iPhone and iPad devices. The prototype is a MAR app that uses face recognition to identify the user and graphically presents related academic information.

KEYWORDS

Information Visualization, Augmented Reality, Mobile, Facial Recognition.

1. INTRODUCTION

The recent technology improvement in mobile devices, as tablets and smart phones, and wireless communication allowed the users to use services as email access, web navigation, social networks and others, that before was limited to the desktop platform, in almost everywhere.

With these improvements, the users demands more and better applications supporting services in mobile devices. It is not limited to the services available in the desktop, but also services that use features of the mobile devices as the Global Positioning System (GPS) for, as an example, acquire information about a facing restaurant, its menu, nutritional information about the food and drinks, etc.

The Augmented Reality (AR) for mobile devices, in association with geolocalization or image recognition, allows new ways to relate information and interaction with the real world. Basically, AR overlays information layers over real images, live or not, catch by the mobile device camera. The commonly ways to present information using Mobile Augmented Reality (MAR) applications are using text, images and video, being the first two, or a combination, of them the most used for the grater majority of the applications.

In this work, we propose the use of Information Visualization (IV) charts into Mobile Augmented Reality environments, as a way to present information enriching the real world vision. The use of IV charts can be used alone or associated to other medias, such as text, images and videos.

This paper presents an architecture designed to aggregate the components and services needed to properly join IV in a MAR environment, detailing its main modules and how it fits together. We also describe a prototype implemented to validate the proposed architecture and a use case of this prototype, offering support to the interaction between teacher and student in a classroom context. The prototype allows the teacher to use the mobile to point the camera to a students' face, recognize it and get academic information about the student (like grades, absences, etc.) and its comparison against the class mediums in a graphical form.

For the prototype development, we choose a client server approach, using iOS 6 in the client side, the PRISMA (Almeida & et al., 2009) information visualization tool and the OpenCV recognition mechanism (OpenCV, 2012) in the server side and access to distributed data sources, locally or on the cloud.

The rest of this paper are organized as follow: Section 2 and Section 3 presents, respectively, some concepts about mobile augmented reality and information visualization; Section 4 list some of the previous works that served as base for this work; Section 5 shows the developed prototype, detailing its architecture and main functionalities; finally, Section 6 make some considerations about this work and lists future works.

2. MOBILE AUGMENTED REALITY

With the appearing of tablets and smartphones, it became possible to run in mobile devices applications that were restricted to the desktop environment, once they have the required resources to run more complex and multimedia applications, resources as more powerful CPUs, more memory, multimedia resources and a considerably amount of sensors (as compass, accelerometer and cameras). So, the concept of Mobile Augmented Reality found a favorable environment, in the current tablets and smartphones, for its diffusion.

The Mobile Augmented Reality generates virtual images mixed to the real scene outputted in the user's video screen. Beyond the characteristics pointed before, the mobile devices had the advantage of being easily carry able, connectable and wireless devices. For those devices the concept of See-Through Video is the most commonly used. The device's integrated cameras catches the environment images in real time and the augmented objects are superimposed to the camera image, creating the augmented reality image that is showed to the user (Figure 1).



Figure 1. AR application with tourist ends, running on iPhone; (a) German Parliament (augmented image overlay) (b) The Sydney Opera (augmented information). (Endigy, 2011)

3. INFORMATION VISUALIZATION

The Information Visualization research area studies the transformation of abstract data into images easily visualized and understood by the human beings (Spence, 2001). Information Visualization tools are computer-based tools that support interaction and data presentation mechanisms. The tools should allow the user to easily and efficiently manipulate and visually rearrange multidimensional data in order to support data analysis. The goal of information visualization tools is to improve data perception, correlation and exploration (Ward & et al., 2010).

The characteristics of a good information visualization tool are defined according to the following user tasks (Carr, 1999).

- Overview of the entire dataset.
- Zoom: focus on a certain data subset.
- Filter: reduction of the dataset according to specified attribute values.
- Details on Demand: additional data provided by an action performed by the user.

In this work, we decide to use the Bar Chart and the Star Chart techniques. The bar chart is a XY chart that shows the relation between amount and value of a given attribute using bars. One axis represents the amount scale for the different attribute values in a given dataset, while the other represents the assumable categorical values for the attribute. Figure 4 shows bar chart.

The star chart is a multi-dimensional technique that represents each attribute in a radius, starting from a common initial point. These attribute axes are uniformly angled and do not necessarily are in the same scale,

as long as a proportion among the scales are kept. Once the axes structure are prepared, for each item in the given dataset are plotted line segments connecting the points related to that item value in an axis pair.

4. RELATED WORK

Butchart (2011) presents a brief comparison among architecture styles for mobile augmented reality using AR browser. Butchart uses a reference model (Reicher, et al; 2003) to classify kinds of architectures: Gateway, Web and Standalone. The Gateway architecture is the approach used for most of AR browsers, being characterized for the need of a web server that attends to the browser requests for content and other information. The Web architecture eliminates de server as centralizer of the requests, making the AR browser access directly the WWW and its services. In the Standalone architecture the application is not dependent of an internet connection having full control of the application data and not needing to handle with data updates, but requiring a full installation of a new version when a data update are required. The Gateway and Web architectures are show in Figure 2.

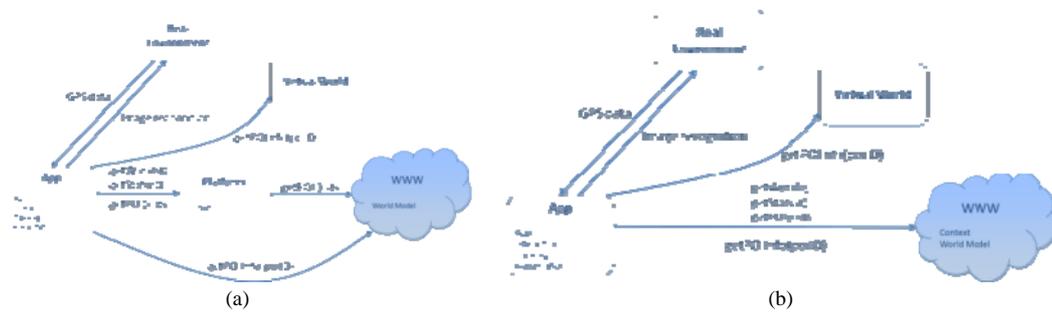


Figure 2. Mobile AR architectures (a) Gateway e (b) Web (Butchart, 2011)

(Dantone, at el., 2011) presented an automatic system for face augmentation on mobile devices (Figure 3). A user can point his mobile phone to a person and the system recognizes his or her face. A tracking algorithm overlays information about the identified person on the screen, thereby achieving an augmented reality effect. The tracker is running on the mobile client, while the recognition is running on a server.



Figure 3. Automatic Face Recognition on Mobile Devices. (Dantone, at el., 2011)

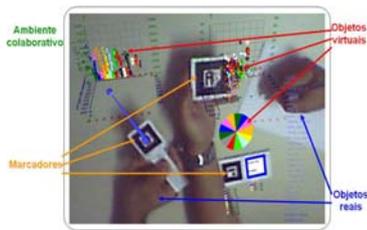


Figure 4. Augmented Reality Visualization Information. (do Carmo, at el., 2007)



Figure 5. Examples of mARGraphy: Dish viewer. Choi, at el., 2011)

(do Carmo, at el., 2007) presented a prototype that implements multiple coordinated views to information visualization in augmented reality environments (Figure 4). A modified version of ARToolKit has been used to create the augmented environment, and the 3D scatter plot technique has been developed to represent the multiple views of data. The coordination between views supported dynamic queries, data selection, and the configuration of views and of the detail on demand feature.

(Choi, at el., 2011) proposed a tool named mARGraphy (Figure 5), which visualizes information based on augmented reality (AR) technology to understand dynamic 3D information. They have used a traditional map viewer application as study case. It recognizes region of a traditional map with object recognition and tracking method on a mobile platform. Then, it aggregates dynamic information obtained from database such as geographical features with temporal changes, and situational contexts.

PRISMA information visualization tool that explores the use of multiple coordinated views (Godinho, et al, 2007) (Figure 6), and it has been developed in Java language. The PRISMA Web version has used standard web development technologies and patterns like Ajax and Java were applied (Almeida & et al, 2009) (Figure 7). The technologies used for the client application include simple HTML pages with CSS and Javascript for manipulation and automatic generation of the interface. AJAX (Asynchronous Javascript and XML) and JSON (Javascript Object Notation) were used for client-server communication. The server generates a view in response to a user request, converts the view into an image and transfers the generated image to the client.

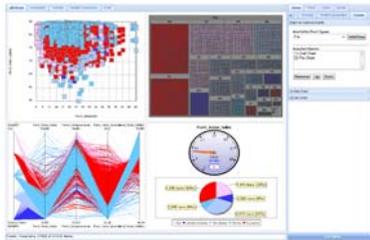


Figure 6. Prisma Web Client. (Almeida & et al, 2009)

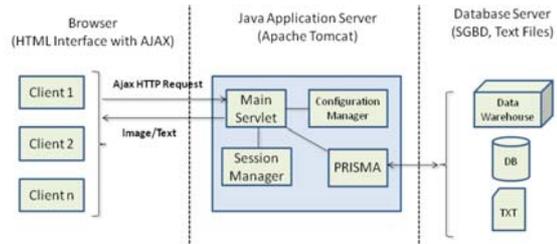


Figure 7. Prisma Web Architecture. (Almeida & et al, 2009)

5. PRISMA - MOBILE AUGMENTED REALITY (PRISMA-MAR)

This section describes the implemented prototype, along with its main functionalities and internal data flow. The architecture defined for the prototype, and the components that compose it, are also presented.

5.1 Architecture

The prototype architecture is client-server based, where the prototype uses the PRISMA Server to handle both the dataset and visualizations processing. The mobile device is mostly responsible for the inputs and outputs of the application. There are two way to input, by the camera and by the display.

Figure 8, shows a macro vision of the prototype functionality, showing an overview of the interaction and communications of each physical agent in the process.

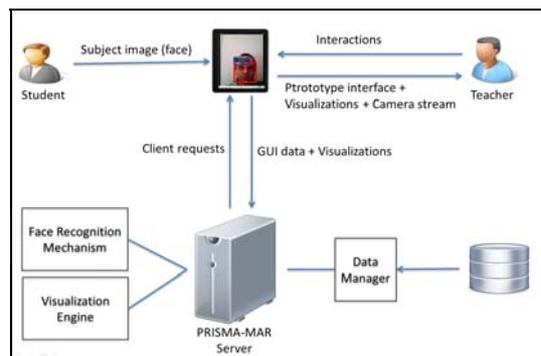


Figure 8. PRISMA-MAR Architecture Overview

The mobile device inputs the video stream (the face to be recognized) and after loading subject data, it controls the interaction from the user. If this interaction causes in the visualization or in the dataset changes (e.g. filters) the prototype sends to the PRISMA Server the new defined configuration and gets the metadata to update the user interface and the new generated visualization.

With the app started, the user point the camera device to the face to recognize, the recognize gets the identifiers of that face and uses to request the data about that subject. Once loaded, the prototype starts the visualization interface, both chart and controls, allowing the user to interact with the visualizations and look for the information needed.

5.1.1 Internal Data Flow

The first user interaction is the login. It not only avoids unauthorized access to students' information, it also reduces the amount of data used in the face recognition process, limiting the search for a face match to the students of a given class.

The Client Core (Figure 9) is responsible for the communication among the client modules and the control of the application execution. The Client Core uses the Facial Tracker to detect a face in the input stream from the camera to detect the presence of a face. Once a face is detected, the Client Core calls the GUI Manager to ask the user if the detected face is to be recognized.

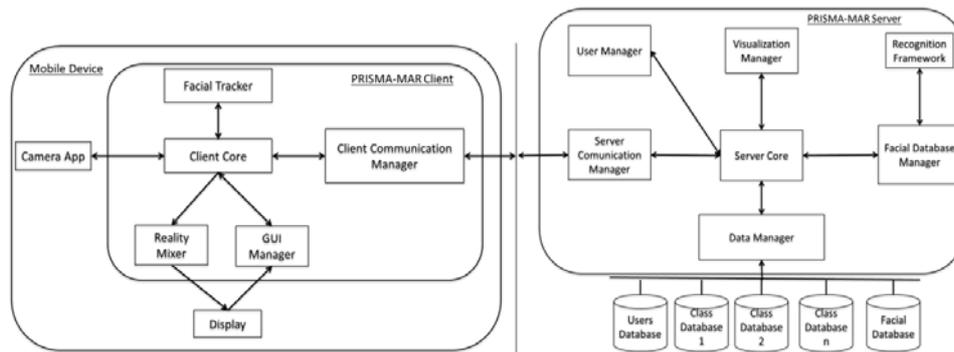


Figure 9. Application flow

The Facial Tracker is a third-part component attached to the prototype architecture as a plug-in. We chose the CIDetector API (Roche, 2011) as this plug-in, but it can be easily exchanged for another facial detection mechanism. It is used only to detect if a face is being caught by the camera and inform it to the Client Core.

If a face is chosen to be recognized, the Client Core gets a set of frames from the camera and passes those to the Client Communication Manager along with a loading request. The Client Communication Manager performs a web request posting the frames with the subject face.

On the server side (Figure 9), the Server Communication Manager receives the load request plus the camera frames, and informs it to the Server Core. The Server Core sends the camera frames and the selected data sources to the Facial Database Manager that uses it to search for a face match and return to the Server Core a subject ID.

The Recognition Framework is a module loaded into the PRISMA-MAR Server. For this initial state of the project, we chose to use as Recognition Framework an implementation of the OpenCV library (OpenCV, 2012), using the Eigenfaces algorithm (Kshirsagar, 2011). The Recognition Framework is used by, and its communication logic, are enclosed in the Facial Database Manager, allowing it to can be replaced with no major code changes in the MAR Server.

The Server Core passes the subject ID along with the selected database reference to the Data Manager. The Data Manager loads the requested information from the data sources (DBMSs, text files, etc.) and processes it to extract the class average for each attribute in the database. The Data Manager then structures this averages as registry and generates a new temporary reduced dataset, containing only the subject information and the class averages, and returns it to the Server Core. The Data Manager also extracts from the database the metadata for the client interface configuration.

The Server Core starts the Visualization Manager with the reduced dataset and the information needed to generate the visualizations (e.g. the type of visualization to be drawn, preferred size of the image, etc.). The Visualization Manager then applies the visual encodings for the reduced dataset and do the math to generate the visualization. After completely generated, the visualization is encoded into an image file format and sent to the Server Core.

Having the visualization image and the metadata for GUI, the Server Core calls the Server Communication Manager to encode and send the information to the client. After receiving this response, the Client Communication Manager passes the return to the Client Core. The Client Core uses the GUI Manager to dynamically create the user interface and fill its controls and visualizations. Once the user interface are completely instantiated, the Client Core sends its information alongside with the camera stream to the reality mixer, for positioning and rendering on display. Once the user interface is rendered, the GUI Manager also

keeps track of user interactions over the display, dispatching those to the Client Core. If the subject is exchanged or the user interacts in a way that modifies the application data or visualizations, the process starts over again. A detailed vision of this flow is shown in Figure 9.

5.1.2 Server Integration

The PRISMA (Almeida & et al, 2009) is an information visualization tool that offers multiple visualizations, support for multiple data sources and an extensible architecture, allowing the use of selected subsets of functionality. Considering the set of services provided for the PRISMA tool and the possibility to develop new interfaces to support new environments over this framework, we developed the prototype server side as a PRISMA interface. The MAR interface abstracts the complexity of handling the data sources, pré-process the data and the process of building the visualization, delegating it to the previously available modules of the PRISMA.

The data and metadata transferred between client and server is JSON (JSON, 2012) formatted and, due to the data reduction step in the server side, it is a pretty lightweight communication (not considering the images). Both textual and images data are exchanged by post requests and responses. At each side (client and server), the Communication Manager holds the intelligence to encode/decode the communication protocol to and from the internal data structures.

The PRISMA-MAR Server provides a one node link for the client, using a single provider for all the kinds of services it may come to consume. This feature does not limit the client, once any third-part service may still be able to load through the server (e.g. a Database Management System on the cloud), it actually provides a safer entry for third-part data (once the MAR Server pre-process the databases and validates it) and makes the client implementation simpler (considering it does not need logic for connecting with different servers).

To the user, this approach reflects in very lightweight client, once it does not include or keep databases, unnecessary code, subject images and any other thing related. It also means improved performance, since the MAR Server performs the databases operations and math processing sending only the displayable results for the client. In fact, the MAR Client is just a bridge (interface) for the user interactions, filling the GUI with the MAR Server responses.

5.2 Application

In the following sections we present the principals characteristics and functionalities of the client-side application.

5.2.1 Login

Beginning the application, the user is prompted with the login screen, requesting its username, password and class, for retrieving data, (Figure 10). After a successful login, the app show a screen with the command bar (initially disabled) containing: Overview, Details and Filter buttons, and a Face Detection Switch component to activate/deactivate the face recognition process (ON/OFF).



Figure 10. Login Screen.



Figure 11. Face Detection Process.

5.2.2 Face Detection

In order to start the detection process, the user should activate (ON) the Face Detection Switch. After that, when a face is detected in the camera image (Figure 12) the app ask the user if that face should be analyzed for further information. In the application context, the face should belong to a student and the information draw as visualization charts are the points of interest in its academic performance.



Figure 12. Face Detected.

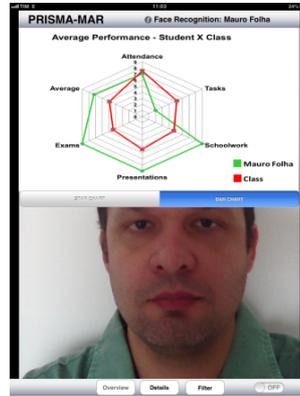


Figure 13. Star Chart Visualization.

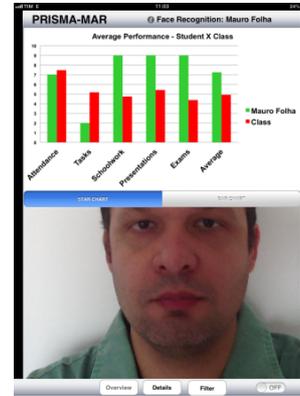


Figure 14. Bar Chart Visualization.

The first graphic loaded to the application is the Star Chart, containing information about the medium of test grades, class attendance, etc. comparing the recognized student's values (green line) with the medium of values of the students' class (red line), as show in Figure 13. The initial visualization is an overview chart about the student's performance and the user can choose between analyze it using the star chart or the bar chart (Figure 14). At this point the Details and Filter buttons are already available for interaction and the Face Detection Switch are disabled.

5.2.3 Details on Demand

In the PRISMA-MAR client, the Details on Demand allow the user to analyze the composition of the subject student media for a chosen dimension. Figure 14 shows a student that got an average grade of 9 in his tests. It does not answer questions as "How many tests did the student took?" and "This student's performance are improving along the semester?". To answer some of these questions, the Details on Demand feature can be accessed by the button Details (Figure 15 and Figure 16), that activate charts about these detailed information, both for the subject student and the class.



Figure 15. Details on demand – Attendance.



Figure 16. Details on demand – Exams.

The access to each dimension are made by the buttons below the detail chart, the user just press the button labeled with the attribute name to be analyzed and that dimension chart are draw in the visualization space. The user can chose the attributes "Attendance", "Tasks", Schoolwork, etc., as show in Figure 17.



Figure 17. Details Choosing Bar

5.2.4 Filter and Zoom

The Filter feature allows the user to insert or remove attributes in the visualization. When activated, the Filter Button render a menu with all the attributes in the dataset to the user select the ones to be rendered (Figure 17). Leaving the Filter menu causes the visualization to be updated, as can be seen in Figure 18, in comparison with Figure 16, with two attributes removed.

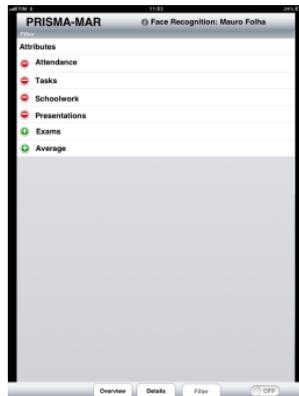


Figure 17. Filter Menu.

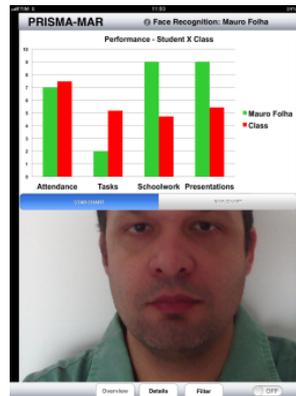


Figure 18. Filtered Visualization.

The Zoom feature was kept as the standard in mobile applications. Sliding two fingers away one to other causes the visualization area to be scaled up, enlarging the visualization.

6. FINAL REMARKS AND FUTURE WORKS

The main contribution on this work is the integration among mobile augmented reality, facial recognition and information visualization. In this paper we presented an application that joining the mentioned concepts, allows teachers to gather academic information about a student and relate it to the average of the student's class.

The main difficulties found during the development of this work were related to found and integrate frameworks of facial detection and recognition. We notice that factors as environmental light, face positioning, occlusion, movement, etc. holds high influence in the recognition process.

We simplified the application defining usual menus as the interaction mechanism, instead of interactions directly over the visualization. The application is also adaptable since the data and visualizations (as images) are sent to the app by the PRISMA-MAR Server. The client application attends to the good characteristics of visualization tool: overview, zoom, filters and details on demand.

The iOS platform was chosen for this first prototype mainly because of the experience of the development group with the platform, but other platforms will be used in future works.

At this point of the development we chose to lock the visualization in the screen once it is loaded to a recognized face, once the face for some reason may be removed of the camera focus. A mechanism similar to a cache memory are currently under development to make the face recognition process more dynamic without overloading client and server with these tasks.

6.1 Future Works

The main future works for this project are listed below:

- Evaluate the user interface and conduct user tests;
- Implement analysis for multiple subjects at time;
- Conduct performance tests in a real environment;
- Improve the level of correct matches in the facial recognition process;
- Design and evaluate a cache mechanism to improve performance in the recognition process;
- Implement new information visualization and coordinated techniques;
- Create new filter mechanisms, for other kinds of data;
- Create an PRISMA-MAR Client for the Android platform.

REFERENCES

- Almeida, L. H. & et al. 2011. Uma Ferramenta Web para Análise Visual de Dados. Proceedings of XXXV Conferencia Latinoamericana de Informática (XXXV CLEI). Pelotas, Brazil.
- Bimber, O. & Raskar, R. 2005. Spatial Augmented Reality - Merging Real and Virtual Worlds. Massachusetts: A. K. Peters Ltd.
- Butchart, B. 2011. Architectural Styles for Augmented Reality in Smartphones. Proceedings of International AR Standards Meeting. Pp 1-7.
- Carr, D. A. 1999. Guidelines for Designing Information Visualization Applications. Proceedings of ECUE'99. Stockholm, Sweden.
- Choi, A. 2011. mARGraphy: Mobile AR-based Dynamic Information Visualization. Proceedings of International Symposium on Ubiquitous Virtual Reality. Jeju, South Korea, pp 37-39.
- do Carmo, R.M.C. & et al. 2007. Coordinated and Multiple Views in Augmented Reality Environment. Proceedings of 11th International Conference Information Visualization (IV '07). Zürich, Switzerland, pp 156 – 162.
- Endigy. (2011). AR in Tourism and Travel. Fonte: Endigy: http://endigy.com/2.5_AR_tourism_travel.php
- Godinho, P.I.A. & et al. 2007. PRISMA – A Multidimensional Information Visualization Tool Using Multiple Coordinated Views. Proceedings of 11th International Conference Information Visualization (IV '07). Zürich, Switzerland, pp 23 – 32.
- JSON. Available in <http://www.json.org/java/>. Last visited in 11/05/2012.
- Kshirsagar, V. P. 2011. Face Recognition Using Eigenfaces. Proceedings of 3rd International Conference on Computer Research and Development (ICCRD). Shanghai, China, pp 302 – 306.
- Milgram, P., Kishino, F. 1994. A Tanonomy of Mixed Reality Visual Display. Transactions on Information and Systems IEICE. pp. 1321-1329.
- OpenCV. Available in <http://opencv.org/>. Last visited in: 11/05/2012.
- Reicher, T. 2004. A Framework for Dynamically Adaptable Augmented Reality Systems. PhD thesis available in: <http://tumb1.biblio.tu-muenchen.de/publ/diss/in/2004/reicher.pdf>. München: Technische Universität München, Universitätsbibliothek.
- Roche, K. 2011. Pro iOS 5 Augmented Reality. Apress – Springer Press.
- Spence, R. 2007. Information Visualization: Design for Interaction. Addison Wesley - ACM Press.
- Ward, M. O.; Grinstein G.; Keim D. 2010. Interactive Data Visualization: Foundations, Techniques, and Applications. A K Peters Ltd.