journal homepage: www.intl.elsevierhealth.com/journals/cmpb

Web-based interactive 2D/3D medical image processing and visualization software

Seyyed Ehsan Mahmoudi[a], *Alireza Akhondi-Asl*[a,b], *Roohollah Rahmani*[a,d], *Shahrooz Faghih-Roohi*[a], *Vahid Taimouri*[a], *Ahmad Sabouri*[a], *Hamid Soltanian-Zadeh*[a,b,c,*]

[a] *Control and Intelligent Processing Center of Excellence (CIPCE), School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran*
[b] *School of Cognitive Sciences, Institute for Research in Fundamental Sciences, Tehran, Iran*
[c] *Image Analysis Laboratory, Department of Radiology, Henry Ford Hospital, Detroit, MI, USA*
[d] *Microsoft Corporation, Redmond, WA, USA*

ARTICLE INFO

Article history:
Received 6 May 2009
Received in revised form
16 November 2009
Accepted 19 November 2009

Keywords:
Web-based software tools
2D and 3D processing and visualization
Medical imaging and analysis

ABSTRACT

There are many medical image processing software tools available for research and diagnosis purposes. However, most of these tools are available only as local applications. This limits the accessibility of the software to a specific machine, and thus the data and processing power of that application are not available to other workstations. Further, there are operating system and processing power limitations which prevent such applications from running on every type of workstation. By developing web-based tools, it is possible for users to access the medical image processing functionalities wherever the internet is available. In this paper, we introduce a pure web-based, interactive, extendable, 2D and 3D medical image processing and visualization application that requires no client installation. Our software uses a four-layered design consisting of an algorithm layer, web-user-interface layer, server communication layer, and wrapper layer. To compete with extendibility of the current local medical image processing software, each layer is highly independent of other layers. A wide range of medical image preprocessing, registration, and segmentation methods are implemented using open source libraries. Desktop-like user interaction is provided by using AJAX technology in the web-user-interface. For the visualization functionality of the software, the VRML standard is used to provide 3D features over the web. Integration of these technologies has allowed implementation of our purely web-based software with high functionality without requiring powerful computational resources in the client side. The user-interface is designed such that the users can select appropriate parameters for practical research and clinical studies.

* *Corresponding author at*: Radiology Image Analysis Lab., Henry Ford Hospital, One Ford Place, 2F, Detroit, MI 48202, USA. Tel.: +1 313 874 4482; fax: +1 313 874 4494.
E-mail address: hamids@rad.hfh.edu (H. Soltanian-Zadeh).
0169-2607/$ – see front matter © 2009 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.cmpb.2009.11.012

1. Introduction

Today medical imaging plays an important role in speed and quality of medical diagnosis. Conventional radiology is prone to problems such as losing films, accessing with delay, spending considerable amount of time and cost to copy and archive the film, and limited application of image processing methods [1]. Towards the end of the 1970s, digital methods for radiology which use digital image representation became widespread [2]. In digital radiology, images are either acquired digitally or converted from analog to digital. In either case, this digital input necessitates strong medical image processing tools to process them.

There are several common functionalities for any medical image processing system. Digital images should be processed, saved, and retrieved easily and quickly using the software. They must compromise their characteristics in terms of reading, writing, and representing different image formats, applying various automatic analysis methods on the images in 2D and 3D, and applying new image processing methods to accurately segment and visualize the data [3,4]. These functionalities are necessary for computer assisted diagnosis and therapy.

Various software for representing, processing, and visualization of medical images have been designed [5–30] including 3D-Doctor, eFilm Workstation, PACSPlus Viewer, EigenTool, MEDAL, Imaris, Caret, Analyze, Vitrea2-Fusion7D, Medx, 3DVIEWNIX, 3D Slicer, Julius, OsiriX, BrainSuite, MIPAV, and MRIcro. Some of these software tools work as image viewers supporting a variety of image formats such as Dicom and Analyze [6,7]. Other tools also provide processing and visualization functionalities such as noise suppression, registration, applying conventional segmentation methods, analyzing images for diagnosis purposes, and representing 2D and 3D data using regular visualization methods. It should be noted that not all of these features are implemented in all of the software tools. Usually, each tool is developed such that it is suitable for some specific applications. For example, 3D-Doctor is developed for processing and visualization, while Vitera-Fusion7D is developed for image registration.

Another important aspect of medical image processing software is the extendibility of the software. As new processing algorithms are developed which are more robust, powerful, and suitable for specific applications, extensions of the software to include these algorithms are inevitable. Among these existing tools, most are not extendible. 3D Slicer is unique as it is open source and the user may add new processing routines to it. This feature has made this software more practical and appropriate for academic and research applications.

An important issue in medical imaging is the ability to access the processed data quickly and easily. Web-based software tools are becoming a popular solution [22–30]. One such example is the Web-Based Multi-layer Visualization System (WBMVS) [30]. However, these web applications either suffer from poor user-interface (UI) functionalities or require the user to install ActiveX or Java Applet components. This causes the loss of many advantages of web-based applications. Using a web application, by centralized storage of data and availability of the software on different platforms, physicians can easily access patient's data from any computer station, regardless of relative processing power or operating system.

There are some fundamental advantages in using web-based medical software. Because of rapid changes in medical tools, the ability to update the software without interrupting the users and modifying their hardware and local software is necessary. Further, there are many complex processing algorithms that require large memory and computational resources that may not be available on many client machines [30]. By centralizing processing power and using server machines with large computational resources, it is possible for the client machine to work as a simple terminal. The acquired data can be saved on the server side and users can access them via the network. The importance of this structure is particularly evident in emergency conditions where quick decisions should be made in locations where resources are limited, such as outside of the hospital or in the field. Physicians can access and process the patient's medical data quickly and easily using the proposed web-bases medical imaging software.

Although there are many Picture Archiving and Communication Systems (PACS) available for centralized data access, most of these systems are designed for Local Area Network (LAN) access, not for internet use. Also, all current web-based PACS systems are based on ActiveX or Java Applet, which are not considered pure web-based solutions, because they impose pre-installation requisites and are not platform independent. In addition, current PACS solutions normally do not perform algorithmic processing on the image and simply serve as image viewers.

In this paper, we introduce a pure web-based, interactive, extendable, 3D and 2D medical image processing and visualization application, which requires no client installation. Our software uses a four-layered design consisting of an algorithm layer, web-UI layer, server communication layer, and wrapper layer. To compete with extendibility of the current local medical image processing software, each layer is highly independent of the other layers. A wide range of medical image preprocessing, registration, and segmentation methods are implemented using open source libraries. Desktop-like user interaction is provided by using AJAX technology in the Web-UI. For the visualization functionality of the software, the VRML standard is used to provide 3D features over the web. Integration of these technologies has allowed implementation of our purely web-based software with high functionality without requiring powerful computational resources in the client side. The UI is designed such that the users can select appropriate parameters for practical research and clinical studies.

The organization of the rest of the paper is as follows. In Section 2, we describe design constrains and technology choices. Section 3 describes our software architecture. Section 4 describes the user-interface of the software. Section 5 compares our software with software tools developed previously for representing, processing, and visualization of medical images. Finally, in the Section 6, we present discussion and conclusions.

2. Design constraints and technology choices

In the design of a web-based medical image processing system, there are a number of design constraints that should be considered. These constraints will limit the technology choices. In this section, we review design constraints and appropriate technology choices.

The first important aspect of a medical image processing system is a user-friendly graphical interface. The interface should allow the user maximum online interaction capabilities. Most of the web applications that use simple web forms do not provide appropriate user interaction facilities. After each interaction of the user, the entire page should be reloaded. This type of the client–server interaction limits the features which could be provided to the user [31]. Therefore, using simple web forms is not appropriate for this kind of application.

A totally different approach in building web-based applications is to develop client applications that should be downloaded and installed on the client machine. The client application can be a separate application or it can be an ActiveX or Java Applet which runs in a browser context. In this case, software platform requisites are imposed on the client machine such as operating system and specific library dependencies. Even for Java based applications that are considered to be platform independent, there should be a JVM installed on the client machine and also there might be some security constrains on the client machine which might prevent the client application to run properly. These restrictions include administrative permissions that are needed for installing new applications or plugins. Such constrains prohibit the user to use the software everywhere on every machine without specific installations. Furthermore, in this case, the application looses one of the main advantages of a web application, which is updatability of the application without interfering with the user. This means that when the application is modified, the user should be informed to install new updates. The point that should be noted here is the technical possibility of updating the software without the client side problems. It is clear that for each update there should be a standard procedure for both informing the user about the update and giving the user control on the update. This means that the user can even access the previous versions of the software concurrently or can switch between the software versions. Because of these shortcomings, client applications cannot be considered purely web-based applications.

Visualization of 3D models is another issue in medical processing systems. The user should be able to construct 3D models and visualize them. This task is one of the most challenging parts of the application, because by default, no 3D service is provided by the browsers [32]. A universal and standard solution for this part of the application is considered to be a strong requirement of the application. Another point about the visualization feature is that the user interactions with the visualized 3D model should be handled on the client side; otherwise, the communication traffic of the client–server path prevents the application to work efficiently.

2.1. *Web-user-interface*

Our ideal web application should be instantly available across all types of clients. For this reason, we use JavaScript as the only unique standard in the current state of the browsers that are used as the main platforms of the web applications. Because JavaScript is a scripting language, it can be considered to be platform independent. Almost all of the current browsers from various platforms support the JavaScript routines without any installation or administrative permissions. This makes the JavaScript an excellent choice for handling our client UI.

Our main challenge in using JavaScript is that in most cases it is used only as passive client-side tool which means that there are limited interactions between the client and server. Despite the fact that this makes the page somewhat interactive but in its simple form, JavaScript requires that all information resides in the client side. However, for our proposed medical image processing system, the client must have continuous connection with the server in order to make the application as interactive as possible.

To reach our goals, we combine JavaScript with AJAX technology which uses some potential functionalities of JavaScript. By this combination, our framework's client side web-pages do not need to be refreshed to make a connection to the server and transfer data from the server. As mentioned before, in the classical web form method, it was required to refresh the page in order to send or receive some information from the server. In our designed framework, the client web page can send or receive data and make local changes on the page. This makes the web pages extremely flexible in regards to interactions with the user [33]. We use a standard XML-based communication framework which works based on HTTP protocol to reduce the implementation overhead which makes our communication well-defined. In addition, by using this combination, the security settings on the client machine are solved. Because the communication is done over HTTP protocol, we do not need to setup new security and managing facilities other than the conventional web browsing settings (Fig. 1).

Although the combination of JavaScript and AJAX covers most of the functionalities of our desired medical image processing system, there are some advanced features which are impossible to implement with current standards such as online contrast/brightness tuning and also 3D features needed for visualization. For this reason, we develop an advanced version of the client application with a Java Applet. It should be noted that because of the layered approach used in the design of our software, this decision only affects the UI layer of the software and other parts of the application are commonly used by simple and advanced UI front-ends.

2.2. *Algorithms engine*

The main engine of the software runs on the server. As such, the base image processing functionalities should be implemented on the server. One of the most important issues of the application is the underlying processing algorithms implementation. Because of the layered architecture of the software which is described in the next section, the language and tech-

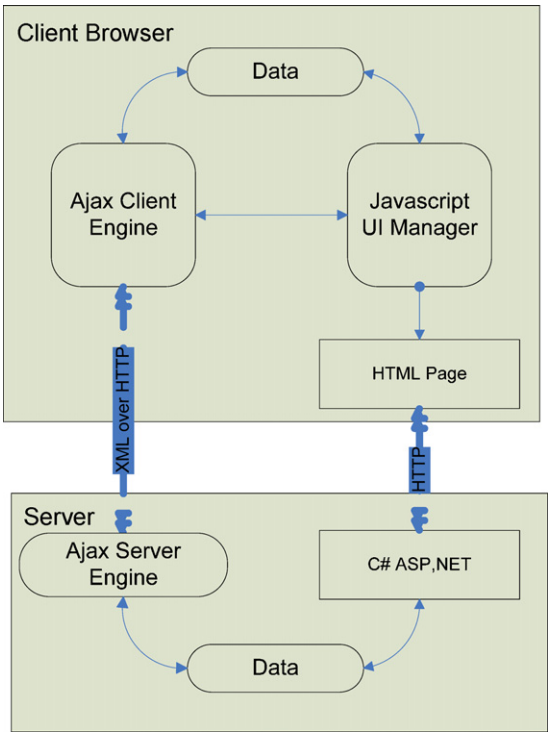

Fig. 1 – The proposed approach for client/server AJAX engine. AJAX works as data communication facility. User-interface management is based on conventional JavaScript routines.

nologies used on the server are highly independent of the UI technologies. So, the technology decision on the server can be done almost independently of the client side. Since most scientific and engineering mathematical codes are written in C++ and there are many efficient numerical libraries, C++ is used as the algorithm engine language. The problem with C++ language is incompatibility of the modern web development languages and conventional C++ code. We solved this problem by using a wrapper layer for the application to allow a high level language (such as C#) communicate with low level numerical routines [34].

There are many numerical and processing libraries available among which the ITK library is used as the base algorithm framework [35]. The ITK library has many low level numerical routines as well as many high level medical processing algorithms, called filters. The ITK library is a well-known C++ open source library designed to cover many medical image processing routines. This library has a template-based structure that supports processing of images with different dimensions. Using the ITK library, the amount of additional code required for 2D and 3D image processing is minimal. The well-designed template organization of the ITK library allows us to write same code for both 2D and 3D image processing. The main ITK strategy for making complex algorithms is using pipelining strategy, which means that the developer can connect a pipeline of filters, whose output is used as the input of the next. This extends the flexibility and modularity of the library and is very helpful for us to develop large and complex applications.

There are many other filters that do not belong to the ITK library but are useful for medical imaging. In the coming section, we have designed a standard for naming convention and a unified model of input and output that allows us to write unified metaphors of coding for the application to reduce the amount of code written by the developers.

2.3. *3D Visualization*

One of the main requirements of our medical image processing system is visualization functionality, where user makes a 3D model and visualizes it. A browser based solution for this purpose is the VRML standard [36], which is a standard 3D world/object definition language designed for use in web applications. This language has its own built-in control library

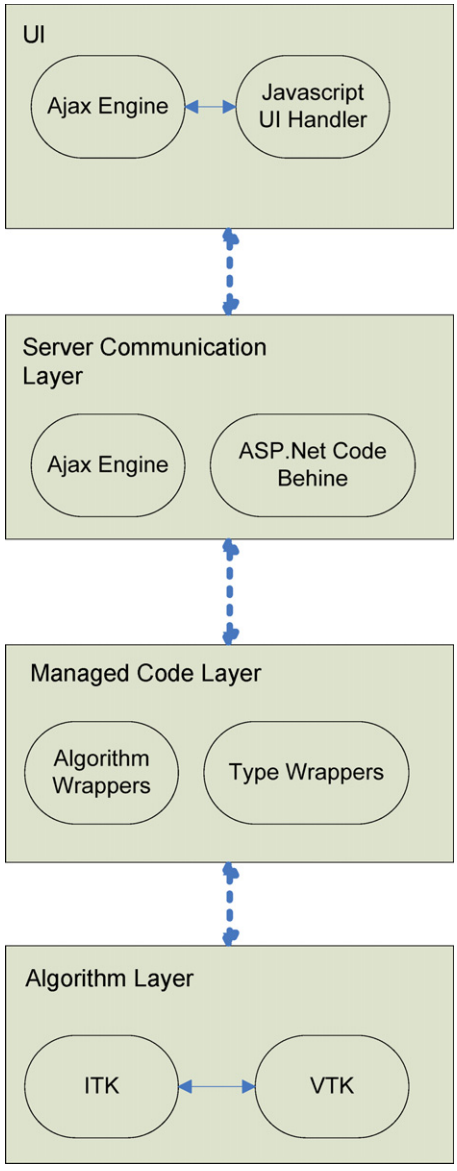

Fig. 2 – Four layers of the software consisting of an algorithm layer, managed code layer, server communication layer, and web-user-interface layer. Each layer has its own sub-components.

and scripting language which makes it flexible enough to be used as the 3D visualization framework of our medical image processing systems. By using this standard, our framework can describe the 3D model built on the server side and sent it to the client. Then, the client user can interact with the model (zoom, rotate, move, etc.) without communicating with the server which increases flexibility and applicability of our framework. The main problem with the VRML files is that the size of the file could be extremely large (30–50 MB). This makes problem for server to handle the traffic and also for clients to download it. For this reason, we use compression facilities for the VRML, which make it possible to send the compressed file for the client. Because the VRML standard is a text-based standard, compression reduces the size of the file greatly (>1–3 MB). This minimizes the amount of network traffic between the client and server.

Another point about visualization is the scripting language that can be used for the VRML which allows construction of a widget library for visualization purposes. By using this library, our framework gives the user maximum control of the visualization process.

It should be noted that despite the fact that the VRML is a universal standard for 3D web features, many browsers do not have the ability to show the VRML files. For this reason, the user should install a VRML viewer plug-in on the browser. There are many different VRML plugins for different platforms and also different browsers. This may seem contradictory against the "No Installation" guideline mentioned in the previous sections. However, this installation is not related to the entire application and only concerns the 3D features. Also, the installed part is not a part of the image processing software in that there is no need to update it.

3. System architecture

In this section, we lay out the design of our software architecture. Most modern web application designs follow the layering principle. In the layered approach, the software is broken to a number of almost independent layers, and only the interfaces of the layers are fixed. This design allows the developers to

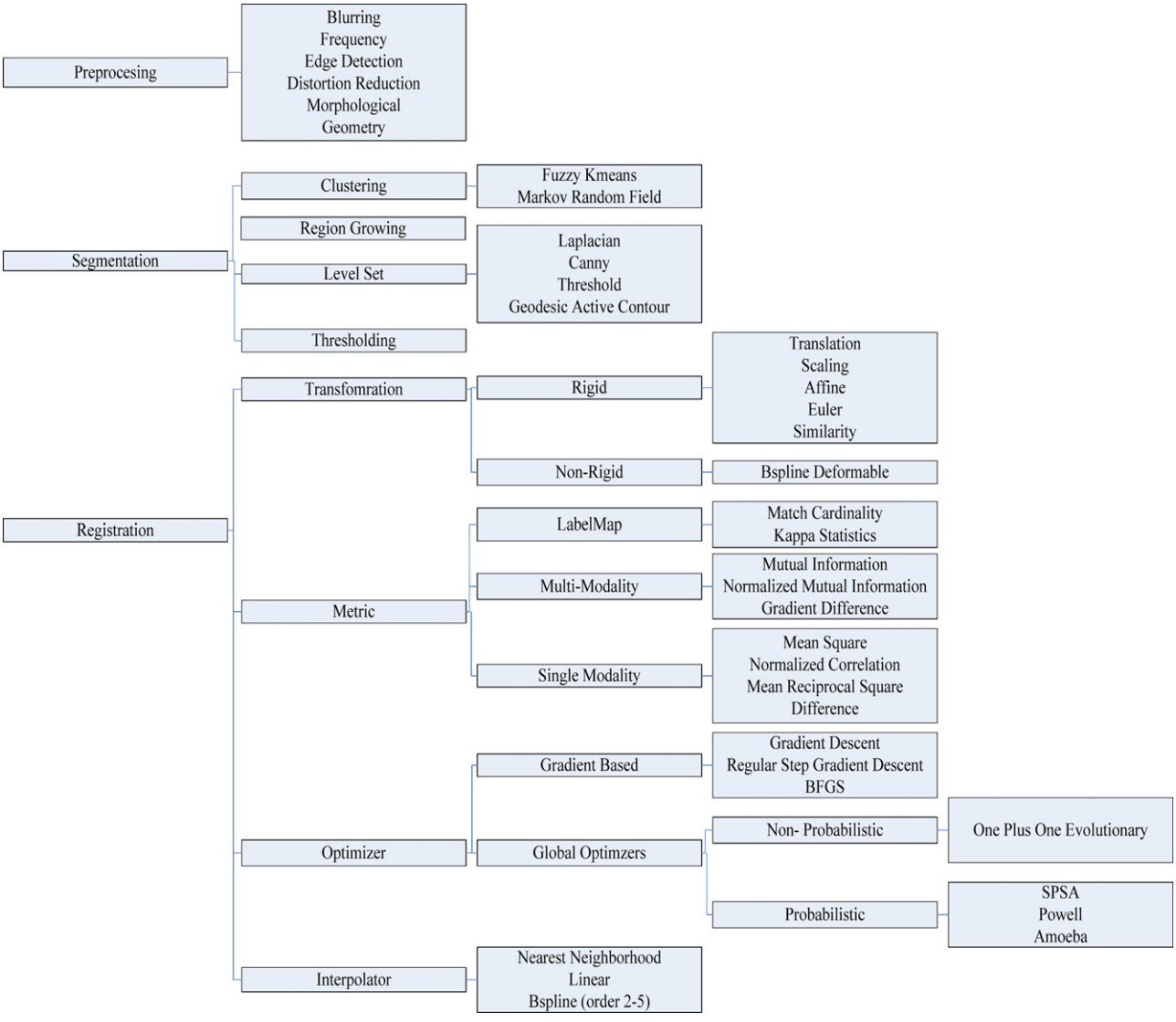

Fig. 3 – The algorithms implemented in the software, categorized in preprocessing, segmentation, and registration modules.

change or modify each layer without changing the other parts of the software.

The most popular approach in web-based software design is the three-layer architecture [37]. This architecture is used the most in MIS applications and reduces the complexity of the software development. In medical image processing applications, there are some additional factors that impose a different layering approach. We designed a four-layer architecture consisting of the UI layer, server communication layer, managed code layer, and algorithm layer (see Fig. 2). We describe our application layers in detail below.

3.1. *UI layer*

The main feature of our UI is its user interaction facility which makes it attractive for commercial and research usages. Traditional web-page models are based on request–response model, which needs to reload the page after each request and thus reduces the interactions of the software with the user. The described technologies allow our framework to change the web-page content without refreshing the page. This expands our UI features significantly. It should be noted that web pages are created in ASP.Net and AJAX is used within these pages.

In this layer, a JavaScript UI engine is developed base on the requirements of the UI. This engine covers the base functionalities of a UI such as menus, toolbars, and image handling routines and drawing functionalities. The engine also contains higher level routines for AJAX communication.

Despite the AJAX flexibility, there are some advanced UI features, which are not needed by regular users. Therefore, if users decide to have advanced features available on the application, they can download a Java Applet to use these advanced features. It should be noted that the Java Applet runs on different platforms. Our specific design allows using this extension without changing other parts of the application. To have this ability, we use the standard that the web-page uses for the Java Applet to communicate with the server. The Applet communication routine uses the HTTP protocol to send and receive data from the server. It should be noted that the application has been designed for both 2D and 3D cases. Different filters and UIs are designed for these cases. In the 3D case, the UI routines are more complicated but most of the functionalities are almost the same.

For visualizing 3D models in the UI layer, there should be a VRML viewer installed on the client application, so there is no special effort by the UI layer to build 3D facilities. Most of the work relies on the server side in preparing the VRML 3D model.

3.2. *Server communication layer*

In this layer, communication mechanisms of the server and the client are handled. They are two kinds of communication between client and the server. First, there is a simple HTTP type communication, which is done by the web pages and the Applet with the server. Another type, which is also done in the HTTP protocol, is the AJAX send/receive channel. This data is sent in an XML format over the HTTP protocol. A web server can easily handle these communications. For this purpose, Microsoft IIS is used as the web server and ASP.Net platform is used to handle web pages. For complex HTTP requests such as AJAX requests, we write new HTTP modules to handle higher

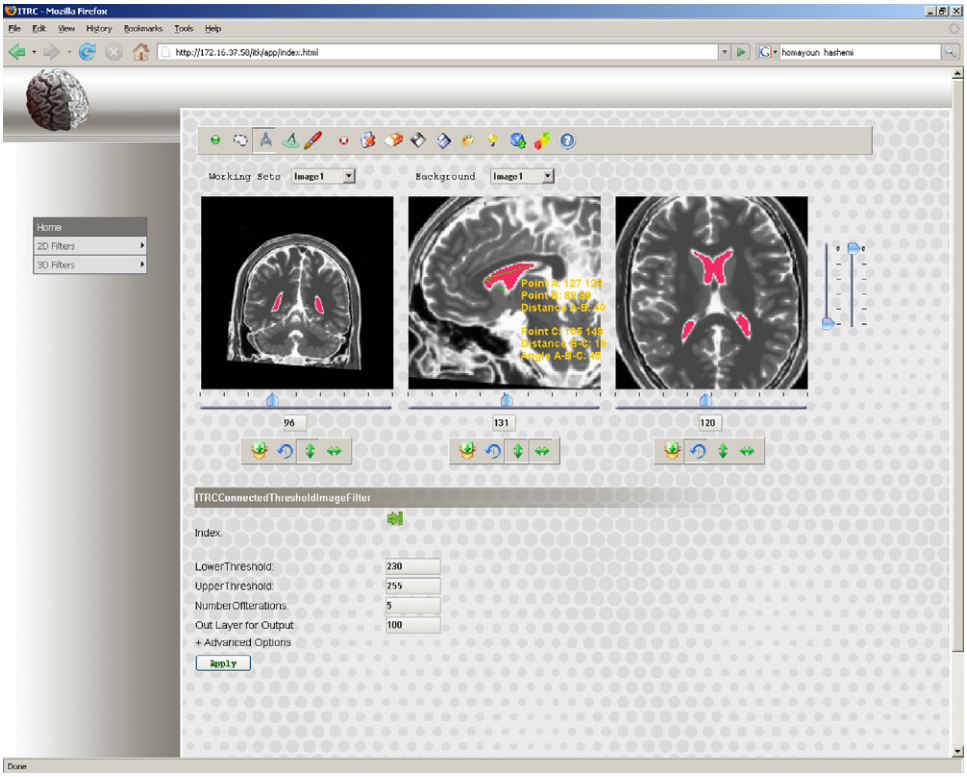

Fig. 4 – A sample result for the segmentation of the lateral ventricles using a region growing method. The zooming and measurement features of the software are shown.

types of protocols over HTTP. This type of communication with the client makes the process more robust and also do not use extra network ports which may have been closed for security reasons.

The code for web-pages also resides in this layer. The web-page codes are written in ASP.Net and C# is used for the code behind language. Because of the facilities of the .Net framework, there is no strict limitation in choosing the language. Another module of this layer is virtual data producers. This module is responsible for building images and sending them to the client.

One of the main concerns about the performance of the software is the upload and download times of the images on the client's machine. Due to the large sizes of 3D images, in this layer an image compression strategy is utilized in order to minimize the number of data packets transferred between client and server. Most medical imaging tasks require high precision and thus we use the lossless GIF format for this task. This significantly reduces the amount of data transferred between client and server. Each image consumes 20 kB of traffic to be shown or updated on the client side. To maintain efficiently in the 3D part of the software, for each slide change in the user-interface only the requested slides are transmitted. These techniques have made the software practical to use over internet links.

3.3. *Managed code layer*

The ASP.Net is used to develop the higher level of the application and also the C++ is used for implementing core processing algorithms. There is no solution for connecting these types of codes directly. Thus, .Net framework is used to provide a solution for connecting conventional C++ to .Net by writing managed code which acts as a wrapper for C++ [34]. Managed code has special rules to handle different issues which may make problem in .Net framework. The managed code layer contains the .Net managed classes which are wrappers for the filters and algorithms written in C++.

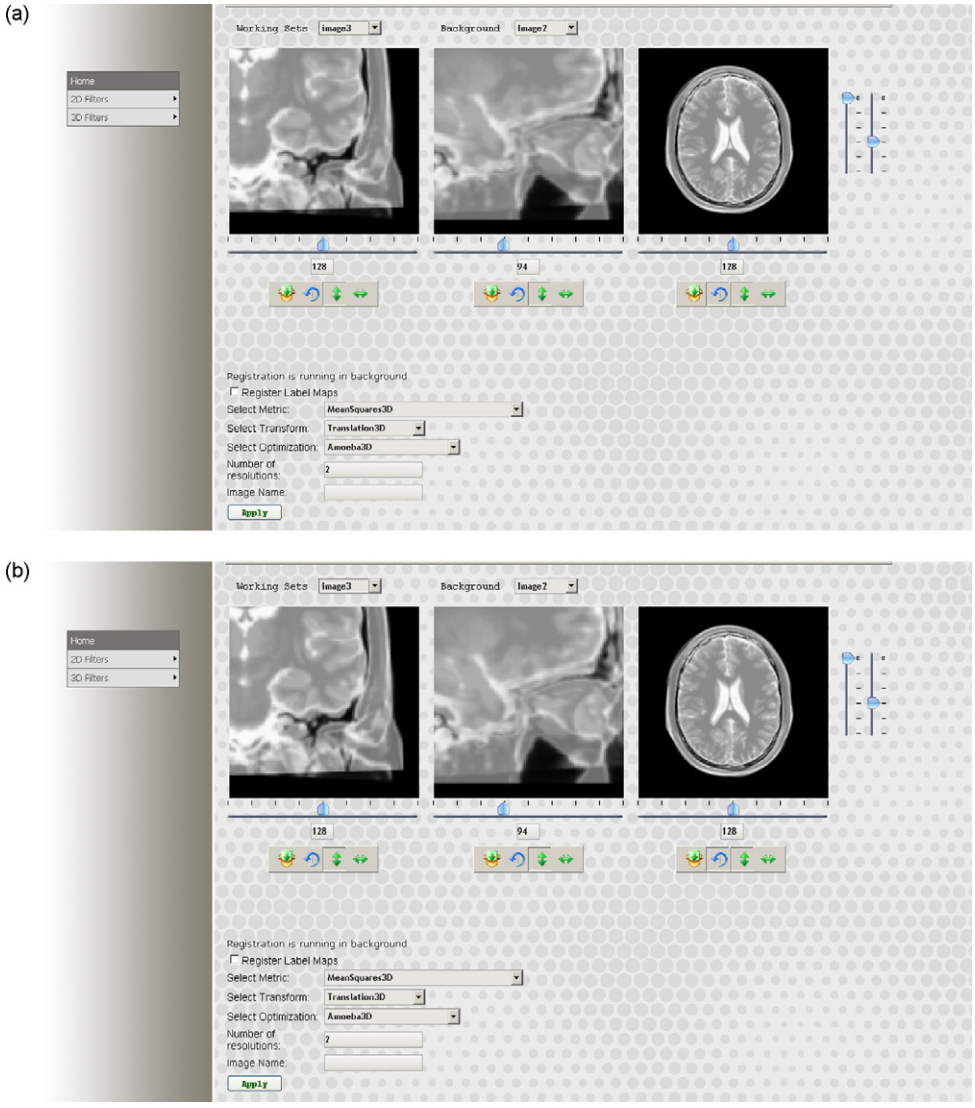

Fig. 5 – The panel for the registration of multi-modality brain images using mutual information as the metric, Powell optimizer, similarity transformation and linear interpolator. The results are shown in three resolutions before registration (a) and after registration (b).

3.4. *Algorithm layer*

The basic layer of our software is the algorithm layer. In this part of the application, all of the processing algorithms are developed. This layer is computationally efficient and also uses efficient numerical libraries provided for rapid development of the new algorithms. As noted before, the ITK library is used as the base numerical and processing library. Many robust methods that are useful for different applications in medical image processing are implemented in the software. These can be categorized into noise and distortion reduction, registration, segmentation, and visualization.

The ITK library is used as the basic framework for adding new algorithms and filters. As it has some filter development coding convention, all the additional filters added to the application followed the convention. This coding standard was also necessary for building a code generator for the software. As described in the next section, a code generator is used for generating most of the codes of the higher layers of the application. For this purpose, the naming and coding of this layer has to follow some strict rules. Based on the fundamental processing algorithms needed for professional medical image processing software and by using the ITK library as the core processing library, many algorithms are implemented (Fig. 3). They include noise and distortion reduction, edge detection, frequency domain low–mid–high pass filtering, region and boundary based segmentation (Fig. 4), and rigid and non-rigid registration (Fig. 5). All of the software tools except for [22] have preprocessing tools with different abilities. Our software has more abilities rather than [5,6,7]. It has preprocessing tools comparable to [8–12] but limited compared to [16]. It should be considered that none of these software tools are web-based. Expect for viewers, all other software tools have segmentation abilities. However, our software includes more segmentation methods than the others except for [16]. In addition, only [16,6,5] have registration abilities but our framework includes more powerful, flexible, registration framework with huge number of available combinations. In addition, our software, [16] and [21] have the extendibility feature. Among the software tools, just [16] has the ability to read 4D datasets.

For developing the visualization algorithm, the VTK library is used, which is a visualization toolkit. The VTK library is widely used in medical imaging applications [38]. The VRML exporting feature of the VTK library has made it a suitable candidate library for the project. It should be noted that the VTK and ITK libraries are highly compatible and thus almost no effort is needed for passing data structures between the two libraries (Fig. 6).

4. 2D and 3D User-interface

In this section, general properties and features of our user-interface are introduced. The software has two general parts which are 2D and 3D image sections. In both of them, a variety of processing methods is available. They can be categorized as noise and distortion reduction, edge detection, frequency domain processing, registration, and segmentation. In addition, 3D images can be visualized and manipulated using visualization section of the application. To maximize user

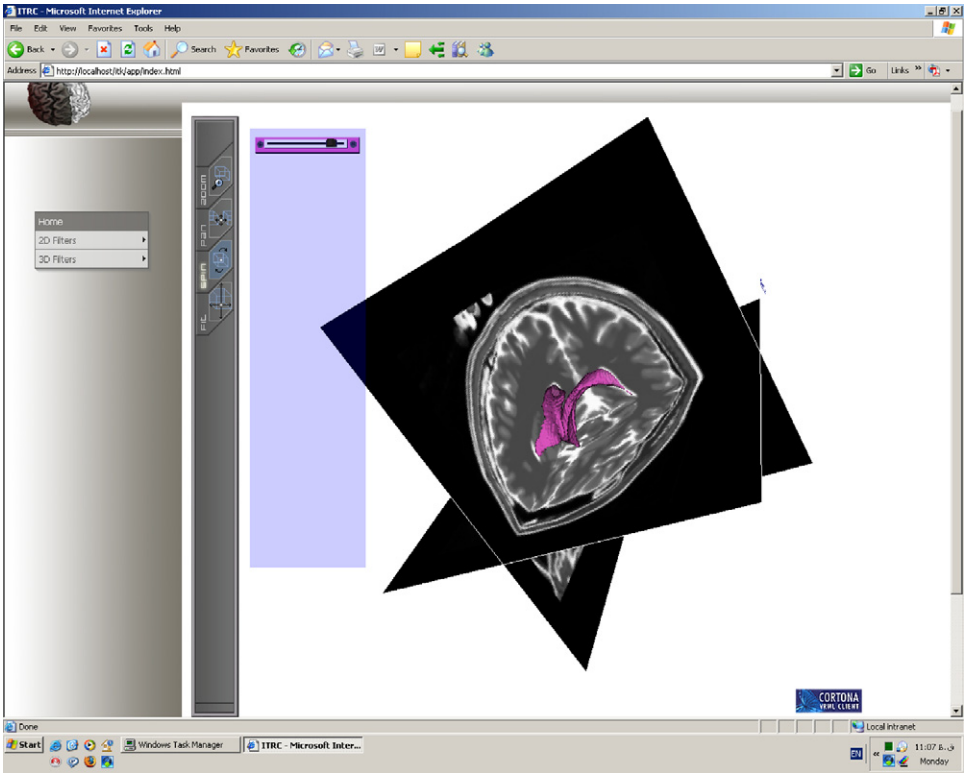

Fig. 6 – A sample of the 3D visualization of the segmentation results. The lateral ventricles segmented in Fig. 4 are visualized in 3D here.

Table 1 – Feature comparison of our software and some popular software in the literature.

	Web based	3D Visualization	Platform independent	Preprocessing	Segmentation	Registration	Dicom support	Extendible	4D Datasets
Our Software	✓	✓	✓	✓	✓	✓	✓	✓	×
3D Slicer [16]	×	✓	✓	✓	✓	✓	✓	✓	✓
eFilm [6]	×	×	×	✓	×	×	✓	×	×
3D-Doctor [5]	×	✓	×	✓	✓	✓	✓	×	×
WBMVS [22]	✓	✓	✓	×	×	×	×	×	×
BrainSuite [19]	×	×	×	✓	✓	×	✓	×	×
MRICro [20]	×	×	×	✓	✓	×	✓	×	×
MIPAV [21]	×	✓	×	✓	✓	✓	✓	×	×

friendliness of the software, different options are added to the application, which can be divided into image handling, quantitative and statistical information, and processing methods parameters. There are a number of features that can be used to compare different software tools. In Table 1, a brief comparison of our software to the other popular medical image processing and visualization software tools is presented. In term of being web-based, there is only one other web-based software that we could compare with ours [22]. This software is for 3D visualization and does not have specific processing abilities. For 3D visualization, our software has more flexibility compared to [5,22] but less flexibility compared to [16] which is due to the VRML limitations of web-based applications. Other software tools do not have 3D visualizations. Our software and [16] are platform independent but others [5,6,19–21] are not.

4.1. *Image handling*

In the 2D section of the software, there is an image viewer that can handle background, foreground, and labelmap images. In the 3D section, there are three image panels that correspond to axial, sagittal, and coronal views. This feature is unique among the software tools introduced in the Introduction Section. More precisely, [16] and [19] only have a 3D section and the others only have a simple panel for image handling. In our design, any of the images in the 3D section can be exported to the 2D section for further 2D processing giving more flexibility to the framework. In either section, the user has a workspace consisting of different datasets. These datasets can be loaded from the server or uploaded by the user. For each image, there is a labelmap attached to it which has intensities between 0 and 255 corresponding to different colors. The users are allowed to select their desired colors. The software has the ability to show foreground and background images with the labelmap using different levels of opacity similar to [16]. Each image can be rotated, flipped or mirrored. Two types of continuous and discrete zooms are available for image viewing. Each image can be viewed separately in a new window. In addition, each image can be zoomed and translated. For the labelmap, the software contains a drawing tool that can be used for segmenting specific parts of images. In addition, the label of any connected parts of the labelmap can be changed, any part of the labelmap can be deleted, and intensity of the foreground image and the labelmap can be seen for particular pixels or voxels. The labelmaps can be saved as new images for further processing.

Some processing methods need a seed point as an input parameter. Thus, an important property of the software is the ability to add/delete/highlight points on an image for the processing algorithms. This feature is similar to that of [16] with a difference that in [16] the user can put points in the 3D view panel. Many image formats are supported in our software for 2D and 3D datasets for uploading (loading) and downloading (saving) of images and labelmaps. For 2D, our software can load and save popular image formats such as tiff, gif, jpeg, and Dicom. For 3D, our software supports the Dicom series and Analyze data sets that are loaded and saved in a zip file (because these formats have multiple files that cannot be downloaded simultaneously). Note that although [5–21] sup-

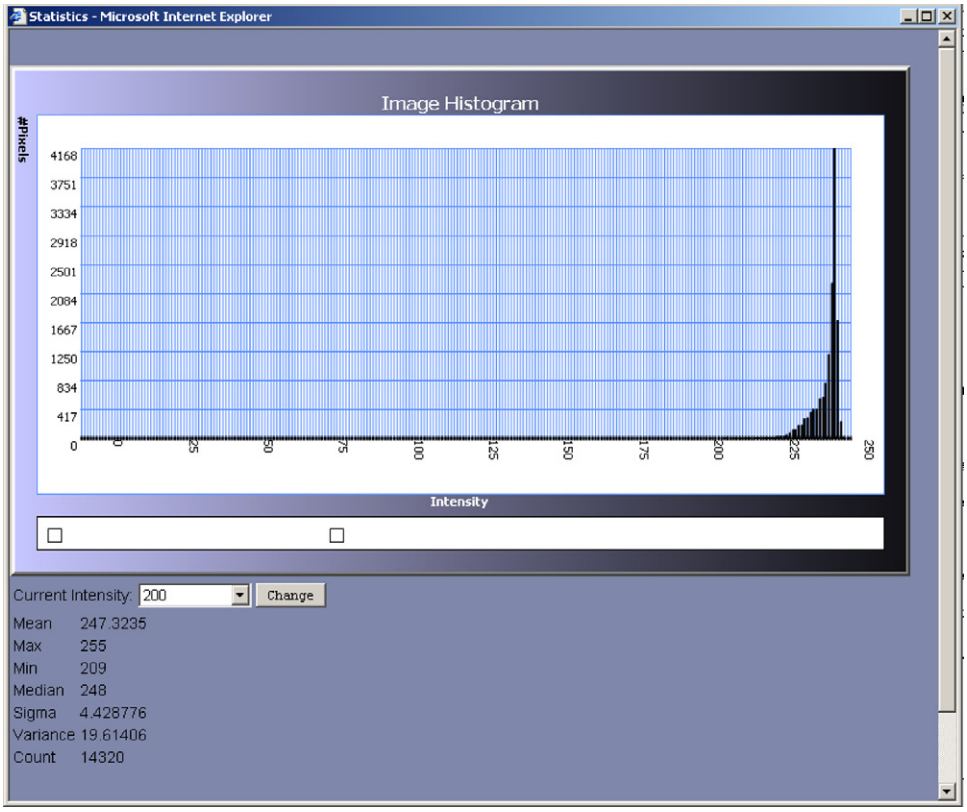

Fig. 7 – An illustration of the extracting quantitative information from a segmented image. The histogram, mean, maximum, minimum, median, standard deviation, variance, and volume of the segmented region are shown.

port Dicom, the web-based framework of WBMVS [22] does not support it.

Since our software is web-based, any processing requests are sent to the server. For making the application computationally manageable, each process is run in the background as a separate thread. This maximizes the computational power of the software and also lets us suspend or stop a process. After finishing an algorithm, a message appears on the page indicating that the results are ready. Also, a list of the processes is available showing their start and ending times and status (any problems, errors, and messages).

4.2. *Quantitative and statistical information*

After processing an image, it is necessary to extract quantitative information from the results. The proposed software has the ability to compute distances between specific points on the images, the angle between two lines defined by three points. It also calculates the mean, variance, standard deviation, surface or volume, and intensity median of a region of the image or the whole image. In addition, it generates a histogram of the pixel intensities in a region or the whole image. An example is shown in Fig. 7. These options are available for both of the 2D and 3D sections. Moreover, other quantitative features can be extracted from the image. In addition, specific filters can be implemented and added to the software to extract desired information from the image.

5. Conclusion and future work

We have developed new software for 2D/3D medical image processing, with minimal traffic between the server and the client. This goal has been achieved using state-of-the-art web development tools such as ASP, .Net, and AJAX. In addition, for making the application platform independent and alleviate the need for software installation on the use computer, we used JavaScript so that conventional browsers can be used as the main platforms of the software. Despite the limitations of the web-based UI, many fundamental features required for medical processing systems have been implemented. They include point selection, manual segmentation, quantitative and statistical information extraction, and image manipulation routines. The software supports many well-known image formats. The software uses a layered design such that each layer can be modified without interfering with the other layers. The ITK is the core processing engine of the application allowing our implementation a variety of processing algorithms to cover a wide range of robust preprocessing, segmentation, and registration methods. For the 3D visualization, the VTK rendering and its VRML exporting facility are used.

The software can be connected to a PACS system to allow direct access to the data. Currently, the data is uploaded to the software for processing and the processed data is saved on client's machine. Because the software supports various image formats including Dicom, it can be easily con-

nected to a PACS server to retrieve and save data on this system.

6. Mode of availability of the program

The Iran Telecommunication Research Center (ITRC) makes the software available to the public. The interested parties may contact the ITRC Technology Transfer Office at tto@itrc.ac.ir for further information.

Conflict of interest

There is no conflict of interest, i.e., none declared.

Acknowledgement

This work was supported in part by a grant from the Iran Telecommunication Research Center (ITRC), Tehran, Iran.

REFERENCES

[1] J.S. Suri, S.K. Setarehdan, S. Singh, Advanced Algorithmic Approaches to Medical Image Segmentation, State of the Art Applications in Cardiology, Neurology, Mammography and Pathology, Springer-Verlag, February 2002.
[2] P.S. Cho, H.K. Huang, Architecture and Ergonomics of Imaging Workstations, A Progress Report USCF Laboratory for Radiological Informatics Research (1998) 151–168.
[3] M. Osteax, A Second Generation PACS Concept, Springer-Verlag, 1992.
[4] F. Chabat, D.M. Hansell, G.Z. Yang, Computerized decision support in medical imaging, IEEE Eng. Med. Biol. 19 (5) (2000) 89–96.
[5] http://www.ablesw.com/3d-doctor/.
[6] http://www.efilm.ca/.
[7] http://www.pacsplus.com/.
[8] http://www.radiologyresearch.org/.
[9] http://www.image.med.osakau.ac.jp/reza/.
[10] http://www.imaris.com/.
[11] http://brainmap.wustl.edu/.
[12] http://www.analyzedirect.com/.
[13] http://www.vitalimages.com/Solutions/Radiology/Functional_Imaging_PET_CT.aspx.
[14] http://www.medicalnumerics.com/products/medx/index.html.
[15] J.K. Udupa, D. Odhner, H.M. Hung, R.J. Goncalves, S. Samarasekera, 3DVIEWNIX: a machine-independent software system for the visualization and analysis of multidimensional biomedical images, in: Annual International Conference of IEEE Engineering in Medicine and Biology Society, 1992, pp. 2082–2083.
[16] S. Pieper, M. Halle, R. Kikinis, 3D Slicer, in: First IEEE International Symposium on Biomedical Imaging, 2004, pp. 632–635.
[17] T. Jansen, N. Hanssen, L. Ritter, B. von Rymon-Lipinski, E. Keeve, Julius—A software framework for rapid application development in computer-aided-surgery, in: Annual Conference of the German Society for Biomedical Engineering (BMT'05), Nuremberg, September, 2005.
[18] http://www.osirix-viewer.com/.
[19] D.W. Shattuck, R.M. Leahy, BrainSuite: an automated cortical surface identification tool, Med. Image Anal. 8 (2) (2002) 129–142.
[20] http://www.sph.sc.edu/comd/rorden/mricro.html.
[21] http://mipav.cit.nih.gov/.
[22] C. Poh, R.I. Kitney, R.B.K. Shrestha, Addressing the future of clinical information systems—web-based multilayer visualization, IEEE Trans. Inform. Technol. Biomed. 11 (2) (2007) 127–140.
[23] P.J. Gage, M.J. Wright, D.K. Prabhu, P.E. Papadopoulos, J. Olejniczak, D. Olynick, Preliminary design interfaces for high-fidelity engineering software: web-based analyses for vehicle engineering, Adv. Eng. Softw. 31 (8–9) (2000) 697–705.
[24] M. Seifert, P. Parkhi, V. Tandra Sistla, K.L. Lawrence, Developing and distributing network based engineering solutions, Adv. Eng. Softw. 31 (7) (2000) 453–465.
[25] J.L. Rogers, A.O. Salas, Toward a more flexible web-based framework for multidisciplinary design, Adv. Eng. Softw. 30 (7) (1999) 439–444.
[26] P. Kwon, H. Kim, U. Kim, A study on the web-based intelligent self-diagnosis medical system, Adv. Eng. Softw. 40 (6) (2009) 402–406.
[27] D. Klimes, M. Kubasek, R. Smid, L. Dusek, Internet-based system for anti-tumor chemotherapy evaluation, Comput. Methods Prog. Biomed. 93 (3) (2009) 292–296.
[28] D.C. Bigler, Y. Aksu, D.J. Miller, Q.X. Yang, STAMPS: Software Tool for Automated MRI post-processing on a supercomputer, Comput. Methods Prog. Biomed. 95 (2) (2009) 146–157.
[29] G.K. Matsopoulos, V. Kouloulias, P. Asvestas, N. Mouravliansky, K. Delibasis, D. Demetriades, MITIS: a WWW-based medical system for managing and processing gynecological–obstetrical–radiological data, Comput. Methods Prog. Biomed. 76 (1) (2004) 53–71.
[30] H.M. Chen, Y.C. Lin, Web-FEM: an internet-based finite-element analysis framework with 3D graphics and parallel computing environment, Adv. Eng. Softw. 39 (1) (2008) 55–68.
[31] E. Dustin, J. Rashka, D. McDiarmid, Quality Web Systems. Performance, Security, and Usability, Addison-Wesley, 2002.
[32] S. Souders, High Performance Web Sites: Essential Knowledge for Front-End Engineers, O'Reilly Media, Inc., 2007.
[33] E.S. Boese, Java Applets (2nd edition) B&W: Interactive Programming, Lulu.com, 2007.
[34] H.M. Deitel, P.J. Deitel, C. Courtmarche, J. Hamm, J.P. Liperi, T.R. Nieto, C.H. Yaeger, Visual C++.NET: A Managed Code Approach for Experienced Programmers, Prentice Hall PTR, 2002.
[35] L. Ibanez, W. Schroeder, The ITK Software Guide 2.4, Kitware Inc., 2005.
[36] A.L. Ames, D.R. Nadeau, J.L. Moreland, VRML 2.0 Sourcebook, Second edition, John Wiley & Sons, Inc., 1997.
[37] W.W. Eckerson, Three Tier Client/Server Architecture: Achieving Scalability, Performance, and Efficiency in Client Server Applications, Open Inform. Syst. 10 (1) (1995) 3–22.
[38] W. Schroeder, K. Martin, B. Lorensen, Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 4th edition, Kitware Inc., 2006.