A software tool for interactive generation, representation, and systematical storage of transfer functions for 3D medical images

M. Alper Selver a,∗, Felix Fischer b, Mehmet Kuntalp a, Walter Hillen b

a Dokuz Eylul University, Electrical and Electronics Engineering Department, Kaynaklar Campus, Buca, Izmir 35160, Turkey
b FH-Juelich, University of Applied Sciences, Medical Informatics Laboratory, D-52428 Juelich, Germany

ARTICLE INFO

Article history:
Received 6 March 2006
Received in revised form
8 January 2007
Accepted 14 March 2007

Keywords:
DICOM
Histogram
Java
Transfer functions
Visualization

ABSTRACT

As being a tool that assigns optical parameters, i.e. color, transparency, used in interactive visualization, transfer functions have very important effects on the quality of volume rendered medical images. However, finding accurate transfer functions is a very difficult, tedious, and time consuming task because of the variety of all possibilities. By addressing this problem, a software module, which can be easily plugged into any visualization program, is developed based on the specific expectations of medical experts. Its design includes both a new user interface to ease the interactive generation of the volume rendered medical images and a volumetric histogram based method for initial generation of transfer functions. In addition, a novel file system has been implemented to represent 3D medical images using transfer functions based on the DICOM standard. For evaluation of the system by various medical experts, the software is installed into a DICOM viewer. Based on the feedback obtained from the medical experts, several improvements are made, especially to increase the flexibility of the program. The final version of the implemented system shortens the transfer function design process and is applicable to various application areas.

© 2007 Elsevier Ireland Ltd. All rights reserved.

1. Introduction

The goal of medical visualization is to produce clear and informative pictures of the important structures in a dataset. Volume visualization is currently in use as a tool to help in diagnosis, surgery, radiological treatment planning, and anatomical education. Therefore, several research activities addressing the limitations of current visualization systems are aiming to come up with new techniques which will carry volume visualization from research and teaching hospitals to routine clinical work. Volume rendering [1,2] is an important technique since it displays 3D images directly from the original dataset and provides “on-the-fly” combinations of selected image transformations such as opacity and color. The only interactive part during the generation of the volume rendered medical images is the transfer function (TF) specification, therefore it is important to design effective and user friendly tools for handling this parameter [3]. Unfortunately, finding good TFs is a very difficult task because of the availability of various possibilities. Since this flexibility cannot be kept in strict bounds, finding an appropriate TF for a meaningful and intelligible volume rendering is very hard.

Current approaches for TF specification can be divided into three groups as manual, data centric and image centric techniques. The Manual approach addresses the need for expert intervention in generating the final image. It states that data

∗ Corresponding author. Tel.: +90 232 4127176.
E-mail addresses: alper.selver@eee.deu.edu.tr (M. Alper Selver), fischer@fh-aachen.de (F. Fischer), mehmet.kuntalp@eee.deu.edu.tr (M. Kuntalp), hillen@fh-aachen.de (W. Hillen).
0169-2607/$ – see front matter © 2007 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.cmpb.2007.03.008
exploration is an essential element of creating the TF if the images were to fulfill the observer’s expectations and to be considered efficient. It is based on the idea that methods which serve to generate images without human interaction would produce nice but ineffective images since they do not consider specific needs of the users [3].

The Data-Centric approaches are based on measuring the dataset properties. Bajaj et al. [4] have used isovalue determination to find the contours that are hidden inside another and show that the isosurface may have more than one component if the isocurves display is associated with a contour tree. Kindlmann and Durkin [5] have assumed that the features of interest in the data are the boundary regions between areas of homogenous material. They used edge detection concepts from computer vision area to define the values associated with these boundaries as opaque. Other data centric techniques use topology analysis [6], stochastic properties of datasets [7], and multidimensional data analysis, i.e. creation of a 3D histogram of data values versus the first and second derivatives [5,8]. The Image-Centric approaches, on the other hand, are based on evaluating TFs on the basis of images they produce where the user can select from among all rendered images presented [9–13].

Currently, neither Data nor Image-Centric techniques are being used in daily clinical work by medical experts because both approaches have some important drawbacks. First of all, TF specification for medical volume visualization is a very subjective task and experts always want to interact with volume data easily and quickly. Since automatic and semi-automatic methods (Data and Image-Centric approaches) cannot take the advantage of user intuition, they leave the user with limited control. In other words, exploring the entire parameter space is no longer possible by using these techniques. Moreover, no proper user interface has been developed for interaction with statistical or metric information provided by Data-Centric techniques limiting their use by medical experts. Image-Centric techniques effectively change the user’s search from an abstract mathematical one to a visually guided one but these techniques have no user interaction. This strategy requires substantial user testing (that has not yet been done) because relieving the user from the data exploration process may be counter productive. Finally, as potentially hundreds of different renderings have to be made, they rely on fast rendering hardware being available to reach their full potential which in its turn increases the costs and vendor dependency.

Because of these disadvantages, the Manual approach is still the current technique in use during daily clinical work. However, manual approach itself is time consuming and difficult, requiring user experience. When there is no prior knowledge about the dataset being visualized, it is hard and tedious to design a TF manually. In this study, a TF editor (TFE) has been developed with a semi-automatic initial TF design method and with several interaction techniques and functions to cover the drawbacks of the Manual approach and other existing programs.

A semi-automatic histogram based method is developed to shorten the design process by creating an initial TF. The advantage of the developed method is that it does not limit the user’s control of the parameter space, creating a good starting point. Moreover several functions are added to increase the user interaction in the design process. These functions are developed to cover the drawbacks of the current TF specification programs. One of these drawbacks is that existing graphical user interface (GUI) designs are not flexible enough [14]. No efficient GUI has been designed which provides the information about the affect of changing a parameter (i.e. color variation) without applying it to the dataset [15]. Another drawback is that a method for initial TF generation to shorten the design process is not available. Rarely, a limited number of predefined TFs are presented. Moreover, there is no systematical storage system for the optimized TF files; only an image-based history tool was implemented [16]. Also, the Digital Imaging and Communications in Medicine (DICOM) standard [17] has never been used to store the TF files. Finally, existing programs are not web enabled and not properly developed for client–server based applications thus limiting their use in teleradiology.

The article presented is organized as follows: Programming properties, plug-in system, and Web based execution are explained in Section 2. The user interface design, the browser to store and access the TF files, and TF editing area are presented in Section 3. Section 4 explains the novel systematic storage system that uses DICOM standard for patient specific storage of 3D images via TF files. The histogram based TF generation algorithm to make an initial design of a TF for a new dataset prior to TF optimization is established in Section 5. Section 6 discusses the optimization of the design by taking into account the feedback from the users. Final discussions and future plans are given in Section 7.

2. Programming

Plug-ins can be defined as a piece of software that can communicate and interact with a host application to provide additional functionality. In this context, the TFE is mainly developed to provide a flexible and highly interactive TF specification interface to the existing visualization programs. It is designed as a software module that can be plugged-into any 3D visualization program that supports Java interface. The “plugging” process is controlled with a simple procedure which consists of the creation of an instance of the TFE, followed by the call of the necessary methods (Fig. 1).

A new instance of the TFE can be created either as an independent frame, which can be adjusted to any size and can be

```java
public void showTFEditor()
{
    // TFE initialization
    TFE tfe = new TFE seriess PANEL THIS, DIRECTORY_SETTINGS_TFE);
    tfe.setDataArray seriess.getSeriesHistogram(0);
    tfe.setDataRangeMin seriess.getHistogramMinX(0);
    tfe.setDataRangeMax seriess.getHistogramMaxX(0);
    tfe.setMaximum seriess.getHistogramMaxY(0);
    tfe.setSeriesInstanceUID seriess.getSeriesInstanceUID(0);
    panelId.setPredefinedTransferFunctions(tfe.getPresets seriess.getSeriesInstanceUID(0);
    tfe.calculateDefaultTransferFunction();
    tfe.updateTransferFunction seriessPanel this.transformFunctionEditor.getActiveTF());
}
```

Fig. 1 – Java code for creating an instance of the TFE.
located anywhere on the screen, or as a fixed panel which is embedded inside the GUI of the host visualization program. This selection is controlled by the first parameter of the constructor method. The second parameter is used to determine the path to locate the predefined TF files. Once the new instance of the TFE is created, ‘setDataArray’ method must be called to send the volumetric histogram data as the input. The other compulsory method to be called is the ‘setSeriesInstanceUID’ method (explained in detail in Section 4). The rest of the input methods (i.e. ‘setDataSourceMin’, ‘setDataSourceRangeMax’, and ‘setMaximum’) are optional and the TFE makes the necessary calculations from the histogram data if they are not called.

The communication between the host visualization program and the TFE is also established with similar methods for sending the output data (i.e. the TF information that the host program receives). The output of the TFE uses two hashtables, one of which consists of the coordinates of the nodes that create the TF versus node colors while the other one consists of the coordinates of the nodes versus their opacity values. The Java class, which sends the output information, extends the native “Observable” class. Whenever the TF information is requested (by pressing “apply TF” button), the host program receives the TF information by calling the ‘getTransferFunction()’ method which in turn calls the native notifyObservers() method of Java and sends the hashtables to the host visualization program. The hashtables are designed in a format that can directly be used in Visualization Toolkit (VTK) [18], which is an open source and a widely used visualization software package. If necessary, the table formats can easily be changed to satisfy the requirements of any other software. To support these changes and future improvements of the developed software, a standard Javadoc documentation is also prepared.

The software architecture of the TFE supports the simultaneous creation of multiple instances. The main advantage of this property is that if the host visualization program supports multiple visualizations in parallel, an instance of the TFE can be created for interaction with each visualization study.

For medical imaging, platform-independent tools, which can easily be transferred and used on multiple platforms are necessary because of the heterogenous environments at medical centers. Since a major claim of the TFE is its ability to be ‘plugged’ to any visualization program, the implementation should be as independent of the operating system (OS) as possible. Application of the platform-independent programming language Java enables the creation of plug-in tools, which can easily extend the basic functions of the systems. Therefore Java is used in the implementation stage of the TFE. In this way TFE runs on almost any OS supporting Java.

When 3D image preprocessing is conducted on an advanced workstation or server, it can considerably reduce the time necessary to achieve the same results on low-cost computers. With the recent advances in network and Internet technology, client–server based 3D image processing systems have become more efficient and popular. However, there is a demand for achieving maximum interactivity, even on low-cost computer side. From this point of view, another advantage of Java is that it can also be used for distributed applications within the global network, primarily for “on-the-fly” extension of the functionality of popular web browsers (i.e. Internet Explorer, Netscape Navigator, etc.). So, the applet version of the TFE, which is executed dynamically by a Java enabled web browser without a need for reinstallation, can be used for the client side of such a system. For example, when the main visualization program is running on a server, the users can use the Internet or any network protocol to execute the TFE and interact with 3D images.

Being Java based, the TFE requires at least a Java Virtual Machine (JVM) 1.4, which needs approximately 100 MB. However if JVM is not installed, a local installation package of 40 MB which contains necessary library files of Java would be sufficient to execute the TFE. When the TFE is running, it requires 30 MB of main memory. The format of the TFE software package is a compressed Java archive (jar) file with a size of 500 KB. The TF files, which are used to store TF information (i.e. shape, color, and opacity), have considerably smaller size (1 KB at most).

3. Implementation of the GUI

During the implementation of the GUI, the requirements needed for TF specification in medical environments are determined by getting feedback from the users (i.e. medical experts) [19]. The following constraints are considered in the design: (1) knowledge and experience level of the medical experts, (2) physical environment of the medical experts, (3) working style of the medical experts, (4) tasks to be performed by the system, and (5) problems that medical experts would like the system to solve. As a task analysis, a detailed list of functional specifications and user interface limitations has been prepared. Moreover, interviews with experts were conducted before and during the design process.

Due to different display preferences of the medical experts (i.e. ordinary, multiple, or wide screen monitors), the TFE is designed so that the user can adjust it to any size that is found proper for interaction. TFE’s user interface consists of six main regions (Fig. 2). This includes the Title Bar, the Menu Bar, the Toolbar, the Browser, the Transfer Function Editing Panel (TF Panel), and the Status Bar.

The Title bar provides the name and directory path information of the active TF file which is displayed on the TF panel. The Menu bar provides pull down menus for accessing the file system operations such as opening, closing, saving, and deleting TF files and folders (File Menu), for displaying labels, grids, texture, and logarithmic histogram (View Menu), and for providing information about using the Browser and the TF panel (Help Menu). The Toolbar displays pictorial representation of different functionalities for the Browser and the TF Panel. The Browser panel provides interactive access to the file system. The TF panel helps in the manipulation and design of the TFs; histogram of the volume data is also displayed on this panel. The Status bar gives information about the last changes made on the TF panel. The user can change the size of these GUI elements relative to each other (i.e. increasing the TF panel area by decreasing the Browser area), which helps to focus on the area that is being used at that moment. Moreover, the Toolbar can be removed from TFE or can be taken outside of the TFE for ease of use.
3.1. **TF Panel and TF Specification**

The TF panel is designed to allow the user to easily manipulate TFs in terms of adding and removing nodes or changing their opacities and colors. A TF consists of two different control points: color and opacity nodes. Color nodes (circle shape) have color and opacity values. Opacity nodes (square shape) have only opacity values. Thus, a color node can be used to change both the color and opacity variations of voxels while an opacity node can only be used to change the opacity variation. Manipulating the TF in terms of adding or removing nodes can easily be done by using a pop-up menu that appears by right clicking over the histogram. By dragging the nodes, the user can change node positions and determine the shape of a TF.

The most important feature of the TF panel is the efficient representation of color and opacity variation of the voxels via an easy-to-understand/use interface. Previously in [15], color and opacity variations are represented on different graphs, which lead to confusion due to the control of two functions instead of one. A disadvantage is that the user cannot see directly if a color will be visible or not (because of the unknown opacity value). In addition, the editing area for each graph is restricted due to the use of two different graphs.

In the TFE, color and opacity change is represented at the same graph by using a color bar and by filling the shapes (circle or square) of the nodes with their own colors. When the user changes the opacity value of a node (by dragging it on y direction), the visibility of that node’s color changes, i.e. it becomes more opaque for larger y values and more transparent for smaller y values. This approach gives the user the ability to change opacity and color values with visual feedback. However it is not sufficient because the user is still unaware of the overall color variation. So, a colorbar is designed and placed at the bottom of the TF panel (Fig. 2). This colorbar shows which color corresponds to which Hounsfield (HU) value and visibility of that color. If a color is opaque, it can be seen clearly on the colorbar. As the color of an HU value changes to transparent, the background texture of the colorbar becomes more visible, warning the user of the reduced visibility of the corresponding color. By using the colorbar, the user can see the effects of changing the opacity and color of a node on the visualized image without doing the time consuming procedure of applying the TF to the dataset.

At the background of the TF panel, a histogram plot is given to inform the user about the intensity of HU values in the volume. It should be noted that, the histogram does not help the user to see which part of the volume image will be affected by that color; instead, it gives important information on how many pixels will be affected due to a color/opacity change.

3.2. **The Toolbar and its Functions**

To cover the shortcomings of the manual approach, several functions and options have been developed based on the experiences of medical experts. The Toolbar provides easy access to these functions, some of which are explained below:

Some standard shapes (i.e. Step, Ramp, Triangle, and Pulse) are the basic forms of the TFs. Since starting with one of these shapes significantly reduces the design time, a toolbar option is provided to allow the user to insert one of these four default TFs. When many of the tissues lie in a very narrow range of HU values (i.e. soft tissues as white matter, gray matter, and CSF), it can be hard to put the nodes of a TF to the exact desired positions. Thus, another toolbar option provides a dialog box, by using which the user can change the HU or opacity value of a node directly by filling the corresponding fields with numerical values. Another option allows the user to adjust the range of the histogram by using the pointers shown in Fig. 2.
is especially useful when dragging a node to an exact position but is difficult due to the need to deal with a range that consists of several nodes (Fig. 3).

Usually the users want to change a part of the TF that consists of several nodes without changing their relative positions to each other. “Region of Interest (ROI)” option gives the opportunity of selecting the nodes in a ROI and moving/scaling them (Fig. 4).

A TF can be applied to a volume by using the “Apply TF” button. The synchronous mode, which is designed for powerful computers, applies the TF automatically whenever the user changes a parameter such as dragging/inserting/removing a node or changing the color/opacity of a node.

Another option allows the user to see the histogram and TFs in logarithmic scale. This option is useful in two cases: (1) when the histogram has strong peaks that suppress the visibility of the rest, the logarithmic view of histogram allows the user to see the histogram in detail; (2) when rendering the datasets with relatively large regions of uniform density, the resulting images are most sensitive to detailed changes in the TF when the opacity is set nearly to zero [20]. Editing the TF within the lowest 5–10% of the range mostly results in the most visible differences in the rendered images, allowing the user to differentiate finer structures in the data. Larger values correspond to renderings that appear nearly opaque, thus obscuring large portions of the present finer structures. However, for smaller or thinner regions, it is sometimes necessary to use higher opacity values to make the region to appear in the rendering. So, a better view can be obtained by scaling the vertical axis of the histogram graph, while the horizontal axis scale is fixed. If the transfer function opacities are scaled logarithmically, the result is a graph in which the regions of intensity are better distributed without pushing any of them off of the graph (Fig. 5). Consequently, it would be easier for a user to precisely control the intensity of a region in the volume.

Finally, the undo button can be used to undo a manipulation step when necessary.

3.3. The TF file system and navigation with the browser

As previously discussed, TF specification is a very time consuming and tedious task. During the TF specification process, a user iteratively explores a very large space of TF parameters. Moreover, the time to optimize a TF depends strongly on the experience of the user. Therefore, saving an optimized TF file (for a patient or a study) and using it later for similar cases can save time and increase efficiency. Also the TF files which were previously found useful, guide the users at finding an appropriate TF. To store TF information, TF files are created, which can be processed (i.e. opened, closed, copied, renamed, and deleted) directly by using the OS or the menu bar. When the TFE is first installed, a directory with the name ‘TF’ is created under the user’s working area. Preset TF files, which were previously designed for CT and MR, are located under this folder. The user can use these default TF files or can create new ones and save them.

Without any navigation system, it will be hard to search for the previously saved TF files as the number of stored files increase by time. Moreover, the user should be able to see and reach the presets and previously saved TF files in parallel with editing. Taking into account these facts, a new browser
is designed to access the file system and store the TF files systematically (Fig. 6). All TF file manipulations as well as OS operations can be done using the browser. If the OS operations are used for editing the files, the browser can be updated by using the browser-refresh button. Also at each execution of the TFE, the dynamic tree structure of the browser checks the changes in the directory and updates the browser automatically.

The file access system provided by the browser helps the user on the systematical storage of TF files and, with this opportunity, the browser can also be used as a history tool. This is explained in more detail in Section 4. Moreover, since every logged in user automatically uses the directory that belongs to his/her account, he/she can store his/her personal TF files. Even when all users use the same account, they can create their own directories and store the TF files.

4. The representation and storage system

As mentioned in Section 3.3, once a good TF is found for a study, it forms a good starting point for the same type of studies and can be optimized with little effort. Therefore, the storage and representation of previously optimized TF files are an important factor that can shorten the design process. Inspired by the Gray Scale Softcopy Presentation State (GSPS) for 2D images, the TFs are used to store and represent 3D images. GSPS objects are separate series in DICOM study in which source images are located only via references, i.e. they are not copied. This way, source images always remain unchanged and multiple presentation states can be applied to the same image. GSPS stores all relevant information about the visual representation of a DICOM image in a separate DICOM object. To describe how an image should be presented on a softcopy display, GSPS objects precisely define all necessary image processing steps.

As GSPS objects are giving the possibility to store and distribute the presentation of a 2D image between softcopy devices efficiently, the same method can also be applied to the 3D images via appropriate TF files. There are two types of TF files for storage: “Global” and “Patient specific”. “Global” TF files (Fig. 6) are valid for all studies of the same kind (i.e. abdomen, head, and brain). The user can give any name to these TF files and can save them under any folder.

"Patient specific" TF files (Fig. 6) are specially designed to be valid for only one DICOM series, which also means for only one patient. For 'Patient specific' saving, the main visualization program must, in addition to the histogram data, send the Series Instance Unique Identifier (SI-UID) (see Appendix A) of the visualized DICOM series to the TFE. This can be done by using the ‘setSeriesInstanceUID()’ method (Fig. 1) when creating an instance of the TFE. If the user selects 'Patient specific' saving, this SI-UID number is saved into the TF file. When the same DICOM series is visualized the next time, the TFE searches the TF directory for the folder with the same SI-UID number. If such a folder is found, the previously saved “Patient specific” TF files in that folder appear in yellow color at the bottom left corner of the 3D screen and can be applied to the volume data with a single mouse click only (Fig. 7).

This storage system approach has three important advantages: (1) Adjustments performed on 3D medical data during diagnosis can be stored as TF files instead of saving the larger size 3D images to PACS, which will in fact only be a duplication. When a 3D image is reconstructed, this object can be used to apply a previously saved TF. (2) With this system, the user does
not have to spend time to see the 3D images which have been previously found and used. Moreover, the system automatically warns the user by showing the previously saved ‘patient specific’ TF files with a yellow label next to the 3D image. (3) TFs are very small in size; therefore, they are inherently well suited to 3D teleradiology applications. For instance, the image series can be transferred for once at the beginning of the teleradiology session and only the TF files can be exchanged online.

5. TF initialization

Automatic generation of an initial TF design is a very critical step, especially when dealing with a new dataset. For instance, an optimized TF for a 3D image obtained from an abdominal CT series is also useful for another series of abdominal CT images. At least, the existing TF provides a very good starting point and can be optimized with little effort. However if a new dataset is being visualized, it is difficult to start the design of a TF without any initial basis. Therefore, a semi-automatic TF generation algorithm is implemented in the TFE. The algorithm is histogram based and uses expert knowledge.

![Fig. 7 – Patient based storage of 3D images via TF files for the same dataset: (a) the TF optimized for heart and bones and (b) the TF optimized for lungs and airway trees. The labels at the bottom left warn the user about the TFs that are previously designed for the dataset.](image)

In medical volume visualization, one advantage is that the structures to be visualized are known to exist in a specified range of gray values (i.e. HU values in a CT series). For example, in an abdominal CT series, the structures of interest are mostly the kidneys, aorta, and liver, but not the skin or fat. It is known that these tissues lie in a known (even roughly) range of HU values. In the developed algorithm, this knowledge is used. First, the user enters the HU range (in CT) or gray value range (in other modalities) for tissues of interest and then selects a color for each tissue of interest. Next, the histogram is smoothed with an averaging filter (Fig. 8a) and peaks are found by detecting the positive to negative crossings of the first derivative of the volume histogram. If a peak is inside the HU or gray value range of a tissue, then the range between the first negative and positive crossings before and after the peak of the derivative of the volume histogram is assigned to that tissue. The assigned tissue is represented using a trapezoid containing one color and three opacity nodes. When the assigned ranges of the two tissues overlap, then the last opacity node of the first tissue is placed at the intersection point with an opacity value of 0.3. If a peak is close to the specified range(s), it might suppress other peaks. Therefore if no peak is found in a specified range, a Gaussian, centered at the peak and with a variance equal to the peak variance, is fitted to the histogram (Fig. 8b). Then the difference between the histogram and fitted Gaussian is calculated. The same peak search is then applied to the residue signal and this goes on until a tissue is assigned within the selected range (Fig. 8c).

Initially, the color variation of the initial TF is set starting from the color of the first tissue up to the color of the last tissue. Then the opacity values of all nodes are fixed to 0.3. Of course, the user can change the color variation and opacity values of the nodes to optimize the TF.

The results for a sample dataset are presented in Fig. 9. The visualized image series are taken from an abdominal CT image series. The tissues of interest are selected to be the kidneys, bones, and aorta. The initial TF and corresponding rendering results can be seen in Fig. 9a and b. Fig. 9c shows the TF optimized by an expert and Fig. 9d shows the results of rendering. Although there is a clear difference between the initial TF and optimized TF, the initial TF design provides a good starting point for the expert.

The proposed method can be used for images derived by all modalities. However, it is most effective in CT volumes because of the well known HU values of the tissues. In other modalities, knowledge and experience of the expert is critical since the gray value range of a tissue should be known to properly design the initial TF. By using this approach, several TF files for different datasets, including CT Abdomen, CT Lung, CT Aorta, CT Neck, CT Head, and MR Brain, have been prepared and optimized (Fig. 6). These TF files provide good basis of initial TFs and thus included in the software of the presented system.

6. User evaluation

For testing the TFE, it is plugged into exploreDICOM [21], which is a dedicated medical image viewer. Twelve medical experts are asked to test the TFE and fill an evaluation form. These
experts are affiliated with Dokuz Eylul University Medicine Faculty Radiology Department, where a complete digital radiology system environment (i.e. Radiology Information System (RIS), Picture Archiving and Communicating System (PACS), diagnostic/clinical workstations, and web viewers) has been in use for several years. 3D visualization programs are routinely in use for diagnosis and treatment planning in the department. Especially, medical experts who work on the images of the brain and abdominal regions use the 3D technology more frequently.

It is determined that the experts joining this study are using 3D visualization from 1 year up to 7 years with a frequency of 5–12 times a month. Twelve questions are asked under four headings and grading was from 1 to 5 where 1 is the best. The views of the experts and the average evaluation value (AVV) for each heading are as follows:

1. **GUI and ease of use**: The experts find the TFE GUI elements easy to use and understand (AVV:2).
2. **TF Panel and specification properties**: The menu options are found to be sufficient. However it is pointed out by the experts that there is a strong need for an information panel which interactively shows the HU values of the tissues. An example of this information panel may be a “mouse listener” which shows the HU/gray value of a pixel that the mouse cursor points on. Such information is indicated to be necessary to give a coarse idea to the user on where to locate the high opacities to visualize the tissue of interest. The visual feedback and manipulation properties of the TF are found to be acceptable (AVV:2).
3. **The Browser and the file system**: The usability of the Browser, TF file format, and its properties such as the history tool are found out to be very helpful. Default presets for different studies and modality types are found to be properly designed; however, small adjustments are still required as the imaging modality is changed (AVV:1).
4. **Patient based storage and representation**: The approach is found out to be very useful. Nevertheless, although the patient based storage of the TF files is found to be very effective, the experts prefer a system that stores TF files to the PACS, not to the computer where the software is running (AVV:2).

These evaluation forms show that the initial results are very promising and optimistic for the TFE. Feedbacks from medical experts show that the TFE is very useful for interaction with visualizations and for producing informative images especially for CT and angiographic datasets. It is pointed by
the experts that it would be more efficient to use the TFE after a segmentation process which eliminates the unnecessary information from the data, especially in MR series where soft tissues are overlapping. For instance, after segmenting the liver from other abdominal organs, it is possible to classify liver tissue, vessels, and tumors by assigning different color and opacities using transfer functions [22].

Another test is made to measure the time needed by both inexperienced and experienced users to define an accurate TF. For this test, three experienced and six inexperienced users are presented with the following three different studies: CT Skull (easy to classify), CT Abdomen (hard to classify), and MR Brain (very hard to classify). These studies are selected by the experts due to their different levels of complexities. The organs of interest, the time needed to classify the studies for the first time with and without proposed TF generation method, and the time needed to classify the same type of studies after the first time, are presented in Table 1. The results are calculated by taking the average time of group members in each category.

7. Results and discussion

A transfer function editor (TFE) has been developed for medical volume visualization. Limitations of the TF specification, especially spatial information drawback, have been overcome by taking into account of HU values and expert knowledge, and using a histogram based method. A user friendly and easy to use GUI has been implemented for fast adaptation to the software. A file type has been implemented that can be used for OS and teleradiology applications and for the storage of 3D medical images using the DICOM standard. A history tool has been constructed to access these files. In addition, the extended TF design functionalities of the developed software ease the TF design process.

The results show that the proposed method decreases the time needed to find an accurate TF by almost a half in CT studies. It saves less time in MR studies because there is no standard measure like HU values used in CT studies. Therefore, the users give approximate gray value ranges for the tissues as the priori information. It can also be observed from the results that once a proper TF is found for a study, recalling it by using the TFE for the same type of studies would reduce the optimization time significantly.

The flexible and platform independent design of the TFE allows the users with different display preferences, distinct experience levels, and with different hardware platforms to use the program easily. The new method of initial TF generation shortens the design process and provides more guidance than including only the presets as the current editors do. DICOM based TF file and 3D image storage system is novel and less time consuming than other image-based history tools.

In conclusion, the developed TFE is found to be helpful and usable in clinical 3D visualization analysis. It is currently in use in DEU Radiology Department especially for classify-
Table 1: Times needed by inexperienced and experienced users to define an accurate TF

<table>
<thead>
<tr>
<th></th>
<th>CT skull (bones, skin)</th>
<th>CT abdomen (kidneys, aorta, liver)</th>
<th>MR Brain (white matter, gray matter, CSF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>First time without</td>
<td>First time with</td>
<td>After the first time</td>
<td></td>
</tr>
<tr>
<td>proposed method</td>
<td>proposed method</td>
<td>(min)</td>
<td>(min)</td>
</tr>
<tr>
<td>First time without</td>
<td>First time with</td>
<td>After the first time</td>
<td></td>
</tr>
<tr>
<td>proposed method</td>
<td>proposed method</td>
<td>(min)</td>
<td>(min)</td>
</tr>
<tr>
<td>First time without</td>
<td>First time with</td>
<td>After the first time</td>
<td></td>
</tr>
<tr>
<td>proposed method</td>
<td>proposed method</td>
<td>(min)</td>
<td>(min)</td>
</tr>
<tr>
<td>Experienced user</td>
<td>&lt; 5</td>
<td>&lt; 25</td>
<td></td>
</tr>
<tr>
<td>Inexperienced user</td>
<td>&lt; 10</td>
<td>&lt; 22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt; 15</td>
<td>&lt; 22</td>
<td></td>
</tr>
</tbody>
</table>

The presented system can further be improved by adding more features aimed at guiding the clinician without limiting user interaction.

Acknowledgements

The authors would like to thank Prof. Dr. Öğuz Dicle and the Dokuz Eylül University Radiology Department for their contributions on this study. The authors would like to thank the reviewers for their valuable critics for the improvement of this paper.

Appendix A

DICOM Information Object Definitions (IODs), which specify the attributes for each object, are description of entities to which the standard refers. These entities include objects such as Patient and Study. An instance of an IOD refers to an actual object of the corresponding class. Attributes of an IOD are divided into three classes, which specify whether or not their presence is required in every instance of an IOD. Some attributes are compulsory, some are desired, and some are optional. Unique Identifiers (UIDs), which are long numeric strings, are used to identify instances and definitions. Each IOD and attribute has its own UID. In the present study, Series Instance UID, which is a compulsory attribute, is used as the identifier.

References

[10] J. Kniss, G. Kindlmann, C. Hansen, Interactive volume rendering using multi-dimensional transfer functions and


[20] S. Potts, T. Møller, Transfer functions on a logarithmic scale for volume rendering, Graphics, Usability and Visualization (GrUVi) Lab, School of Computing Science, Simon Fraser University.
