High-Quality On-Patient Medical Data Visualization in a Markerless Augmented Reality Environment

Márcio C. F. Macedo,
Antônio L. Apolinário Jr.
Federal University of Bahia - UFBA
Salvador, Bahia, Brazil
Email: marciocfmacedo@gmail.com,
apolinario@dcc.ufba.br

Antonio C. S. Souza
Federal Institute of Bahia - IFBA
Salvador, Bahia, Brazil
Email: antoniocarlos@ifba.edu.br

Gilson A. Giraldi
Nat. Lab. for Scientific Computing - LNCC
Petrópolis, Rio de Janeiro, Brazil
Email: gilson@lncc.br

***Abstract*—To provide on-patient medical data visualization, a medical augmented reality environment must support volume rendering, accurate tracking, real-time performance and high visual quality in the final rendering. Another interesting feature is markerless registration, to solve the intrusiveness introduced by the use of fiducial markers for tracking. In this paper we address the problem of on-patient medical data visualization in a real-time high-quality markerless augmented reality environment. The medical data consists of a volume reconstructed from 3D computed tomography image data. Markerless registration is done by generating a 3D reference model of the region of interest in the patient and tracking it from the depth stream of an RGB-D sensor. From the estimated camera pose, the volumetric medical data and the reference model are combined allowing a visualization of the patient as well as part of his anatomy. To improve the visual perception of the scene, focus+context visualization is used in the augmented reality scene to dynamically define which parts of the medical volume will be visualized in the context of the patient's image. Moreover, context-preserving volume rendering is employed to dynamically control which parts of the volume will be rendered. The results obtained show that the markerless environment runs in real-time and the techniques applied greatly improve the visual quality of the final rendering.**

***Keywords*—Augmented Reality; Volume Rendering; Markerless Registration.**

I. Introduction

Augmented Reality (AR) is a technology in which the view of a real scene is augmented with additional virtual information. Medical AR is a sub-field of AR in which the virtual entity is a medical data. Typically, a successful medical AR environment must support:

1) Volume rendering - when the virtual entity consists of a 3D medical volume;
2) Accurate tracking - for the proper alignment of the virtual object into the augmented scene;
3) Real-time performance - for user interactivity;
4) High visual quality in the final rendering - to improve the user's perception of the augmented scene.

Medical AR is useful to solve the problem of on-patient medical data visualization. In general, patient's anatomical structures are shown on monitors and based on 2D images, corresponding to slices of the 3D data. In this case, the physician has to mentally compose what is shown on the screen to the patient. AR takes over this task of mental mapping by transferring it to a computer. Therefore, the physician is able to visualize, at the same time, the patient and a part of his own anatomy in the display. On-patient medical data visualization aims to improve medical diagnosis, surgical operation, post-operative examination and training.

AR applications can be divided into two basic groups: marker-based or markerless. Marker-based AR uses fiducial markers as a point of reference in the field of view to help the system to estimate the camera pose. These markers need to exist in the real world to be tracked by the application. Markerless AR (MAR) uses a part of the real scene as the marker. Tracking becomes more complex in MAR. However, because artificial markers are not part of the original scene, MAR is desirable in several AR scenarios.

Most of the existing medical AR systems use bulky equipment (e.g. marker-based optical tracking systems) to help in the tracking of the 3D medical data. While they give accuracy in the tracking and positioning of the virtual medical data in the patient, in some applications these markers can be intrusive and the hardware too expensive. To solve these issues, markerless tracking can be implemented taking advantage from the visual features of the scene, such as texture or geometry of the object of interest to be tracked. Currently, the high computational power of hardware, mostly based on parallel computing, like GPUs, allied to low-cost capture devices, allows this solution to be used with enough accuracy and in real-time. In this way, we proposed in [1] a MAR environment in which a 3D reference model of the region of interest is generated and tracked from the Kinect depth stream. From the estimated camera pose, the 3D reference model can be displayed and augmented with the volumetric medical data visualization generated by standard volume rendering techniques. However, the obtained solution needs to be improved in some aspects. The registration between medical volume and patient's region of interest is semi-automatic, where the user must position and adjust pose and scale of the volume. Moreover, the rendering quality is poor. Medical volume is almost superimposed over the patient, which is already known to be an ineffective visualization method in the context of on-patient medical data visualization [2], with some exceptions for specific scenarios [3].

The visual quality of the final rendering (i.e. composition between real and virtual AR entities) is fundamental for the success of the application. Instead of superimposing the raw medical volume onto the patient, an improved solution is to extract the region of interest in the volume (e.g. bone, soft tissue, organ) and show it as a focus region in the context of the patient's body. The extraction step can be done by pre-segmenting the volume into regions of interest or by defining a transfer function which highlights the features of volume. Both solutions solve the problem, but they are time-consuming.

In this sense, illustrative context-preserving volume rendering can be applied to define on-the-fly which parts of the volume are relevant and must be rendered [4], [5] in order to reduce the computational cost. The final rendering step based on focus and context regions is called focus+context (F+C) visualization paradigm [6] and it is known to improve the visual perception of the scene in an AR environment [2], [7]. In the medical volume-to-patient registration step, we can reduce user's assistance by scaling the medical volume based on 3D reference model's size. Initial pose can be computed by using a pose estimator for the patient's region of interest. These considerations are the starting point for this work and compose its contributions.

Specifically, this paper presents an extension of the real-time MAR environment for on-patient medical data visualization proposed in [1] in order to address the mentioned limitations. Firstly, a 3D reference model of the region of interest in the patient is generated. Next, the user positions the medical data into the scene based on the reference model. Afterwards, sensor's raw depth data is aligned to the 3D reference model, predicting the current camera pose. From the estimated camera pose, a volumetric medical data can be displayed to a physician at the location of the patient's real anatomy. Volume rendering is performed by standard techniques [8]. Finally, to improve the visual perception of the scene, F+C visualization and illustrative context-preserving volume rendering are employed to give to the physician a tool for direct interaction with the application. Moreover, our approach supports occlusion handling. The final result is shown in the physician's monitor. In the experimental results, we perform the validation of the proposed solution by considering performance and visual quality of the techniques used in this environment.

The rest of the paper is arranged as follows. Section II provides a review on the related work of Medical AR systems for on-patient medical data visualization. Section III presents the markerless tracking solution used in this work. Section IV introduces the techniques used for on-patient medical data visualization, focusing on the volume rendering and its integration with the MAR environment. Section V discusses the experimental results. The paper is concluded in Section VI, with a summary and discussion of future work.

II. Related Work

Medical AR systems for on-patient visualization have been driven by different approaches in recent years. Kutter et al. [7] proposed a marker-based method for real-time high quality on-patient visualization of volumetric medical data on a Head Mounted Display (HMD). Their work focuses on efficient implementations for high quality volume rendering in an augmented reality environment. They also provide occlusion handling for physician hands. An improved version of this work was proposed by Wieczorek et al. [9] to handle with occlusions due to medical instruments as well. Also, they included additional effects in the system, such as virtual mirror and multi-planar reformations.

Debarba et al. [10] proposed a method to visualize anatomic hepatectomy (i.e. anatomic liver resections) in an AR environment. The use of a fiducial marker made possible the positioning and tracking of the medical data in the scene. A mobile device was used to allow the visualization of internal structures of the patient's body.

Suenaga et al. [11] proposed a method for on-patient visualization of maxillofacial regions. A 3D optical tracking system and a fiducial marker are used to track the patient. A semi-transparent display is placed in front of the mouth region of the patient. The display shows the maxillofacial medical data. This method runs in 5 FPS (frames per second).

Different from the previous approaches, Maier-Hein et al. [12] proposed a method for markerless mobile AR for on-patient visualization of medical images. They proposed a system in which a Time-of-Flight (ToF) camera is mounted on a mobile and portable device (e.g. tablet PC, iPad) and the physician might move the portable device along the body of the patient to see his anatomical information. To estimate the camera pose, they use a graph matching procedure [13] and an anisotropic variant of the ICP algorithm [14] to align the surfaces continuosly captured by the ToF camera. This method runs in 10 FPS.

Lee et al. [15] proposed a markerless registration framework for a medical AR system. They use three cameras: two of them are mounted to form a stereo vision system and reconstruct the patient's head; the other camera is used to capture the images of the patient in real-time. In a pre-processing step, a surface is reconstructed from computed tomography (CT) and a variant of the ICP algorithm is used to do the image-to-patient registration. The estimation of the third camera's pose is done by using a fiducial marker.

In the field of anatomy education, Blum et al. [16] proposed the *mirracle*, a magic mirror for teaching anatomy. They used a display device and a Kinect sensor to allow volume visualization of a CT dataset augmented onto the user. To track the pose of the user, they used the NITE skeleton tracking. As the system is for educational purposes, they could use a generic CT volume which was scaled to the size of the user and augmented onto him. Based on the assumption that the Kinect provides inaccurate depth data, Meng et al. [17] proposed an extension to *mirracle* in which landmarks are used to improve the accuracy of the positioning and tracking of the medical data. A recent extension of this application for bone anatomy learning has been proposed in [18].

Mercier-Ganady et al. [3] presented a novel markerless augmented reality application for on-patient brain activity visualization. User's head is tracked by using a face tracking algorithm. Brain activity is computed from an electroencephalography cap (EEG), which is worn by the user.

Most of the approaches described in this section share the

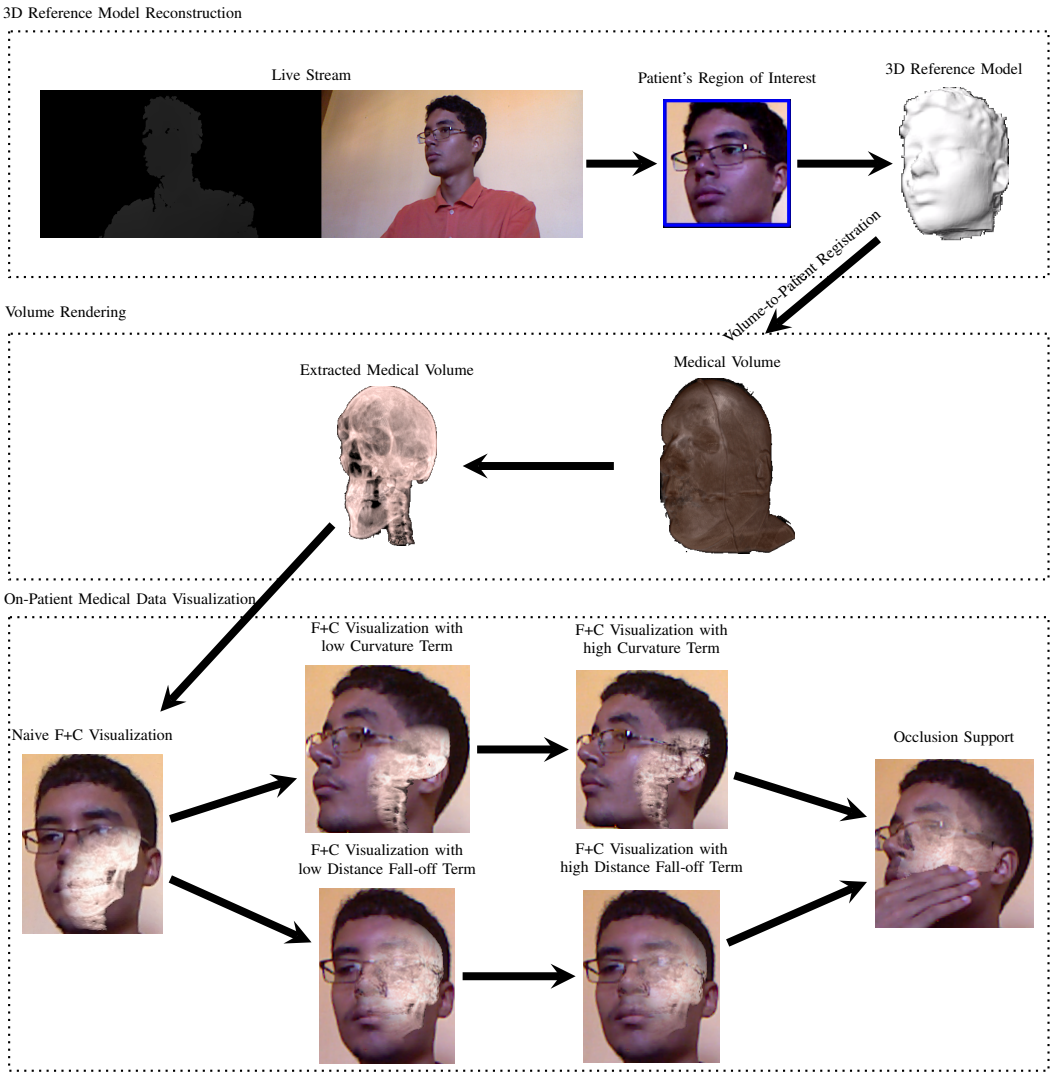

Fig. 1. Overview of the proposed approach. Top layer: From an RGB-D live stream, the patient's region of interest is located and segmented from the rest of the scene. A 3D reference model is reconstructed to serve as basis for markerless tracking. Middle layer: 3D reconstruction is stopped and the medical volume is rendered and segmented by using context-preserving volume rendering. Bottom layer: The user positions the segmented volume into the scene and MAR tracking is done based on the 3D reference model. F+C visualization based on curvature and distance fall-off terms is applied to improve visual perception. The approach supports occlusion.

same drawback: they use markers to help in the calibration, positioning and tracking of the objects in the scene. The use of fiducial markers provides fast and accurate tracking. However, these markers are still intrusive, because they are not part of the original scene. Moreover, the hardware of the optical tracking system in some applications is too expensive. The approaches that does not use marker-based hardware, in general, do not obtain real-time performance in its application due to the computational cost of the markerless tracking in conjunction with the volume rendering techniques used. One exception of this is the *mirracle* system and its extensions. However, the main drawback of the fast NITE skeleton tracking is that it does not track accurately some parts of the body, such as head. Another exception is the approach proposed for brain activity visualization. However, the authors have used a specific technique for face tracking. Different from them, our approach based on a markerless tracking runs entirely in real-time with low-cost hardware components. The solution proposed in this paper is general in the sense that it can track any part of the body with enough accuracy. Moreover, in terms of visual quality, we integrate into our approach state-of-the-art techniques proposed in this field (e.g. [2]) to enhance the augmented visualization of the virtual internal structures on the patient, instead of superimposing the virtual content on the scene, as done by the major of the existing systems (e.g. [10], [12], [15]).

III. Reference Model Reconstruction and Registration

In this section we describe the markerless tracking solution in which this work was based on. An overview of this environment can be seen in Figure 1-top layer.

A. Environment setup

The proposed approach is based on an RGB-D sensor and a computer with GPU, therefore being accessible to any

user. The Kinect [19] is used as RGB-D sensor to capture patient's color and depth information. Real-time performance is achieved by using parallel processing power of the GPU in all critical algorithms.

B. Vertex and Normal Maps Generation

In this subsection we describe the algorithms that are used for every input frame to generate vertex[1] and normal[2] maps, with the exception of the segmentation procedure to extract the patient's region of interest, which is only performed during the reconstruction stage pictured on Figure 1-top layer. All of these algorithms run on the GPU.

As mentioned before, the Kinect has two sensors that capture color and depth information of the scene. Therefore, the Kinect sensors are calibrated to enable a mapping between them.

In order to segment the patient's region of interest from the scene, we need to use a trained classifier or do it semi-automatically. As will be discussed in Section V, our region of interest consists in the patient's head. Therefore, we apply the Viola-Jones face detector [20] implemented in GPU to locate and segment the face in the color image. From the segmented face, its location can be transposed to the depth image by using the extrinsic parameters computed from the calibration step. Doing so, we achieve a more restricted area of the depth map to be used through the other steps of our approach.

Depth map is denoised using a bilateral filter [21] that preserves discontinuities of the raw depth map. Depth of the background scene is segmented by applying a Z-axis threshold on the filtered depth map. This threshold is defined semi-automatically.

Filtered depth map is then converted into a vertex map and a normal map. Vertex map is constructed by the product between filtered depth map and Kinect IR camera's intrinsic calibration matrix. Normal map is constructed by computing the eigenvector of smallest eigenvalue of the local covariance matrix computed for every vertex. This technique produces normal maps with less error than traditional approaches based on neighboring points for normal estimation [22].

C. 3D Reference Model Reconstruction

With the patient's region of interest properly segmented, we need to track it through the Kinect live stream (whose tracking algorithm is described in the Section III-E) and to integrate the different viewpoints acquired from the scene into a single reference model shown in Figure 1.

KinectFusion algorithm [23] is used in this context to reconstruct a 3D reference model in real-time. This algorithm integrates raw depth data from a Kinect into a volumetric grid to produce a high-quality 3D reconstruction of a scene. The grid stores at each voxel the distance to the closest surface (i.e. Signed Distance Function - SDF) [24] and a weight that indicates uncertainty of the surface measurement. SDF values are positive in-front of the surface, negative behind and zero-crossing where the sign changes. In KinectFusion, SDF is only stored in a narrow region around the surface, in other words, a truncated SDF (TSDF). These volumetric representation and integration are based on the VRIP algorithm [25]. Surface extraction is achieved by detecting zero-crossings through a ray caster. All these operations run on the GPU.

The 3D reference model reconstruction is done only one time and is the basis for MAR live tracking. The reconstruction is stopped semi-automatically and the medical volume-to-patient registration can be computed.

D. Medical Volume-to-Patient Registration

The next step is the placement of the medical data into the scene. However, as will be discussed in Section V, in this work the patient's region of interest is augmented with a generic volumetric data instead of his own. Therefore, we do not use an automatic registration method. Instead, we propose a new method for coarse registration between medical volume and patient's 3D reference model. Then, the result can be fine adjusted by the user. An overview of the coordinate systems used for the coarse registration process is shown in Figure 2.

Given the reference model M_{ref} extracted from Kinect-Fusion's volume and the medical volume inside a normalized unitary bounding box, the scale factor S_{med} can be computed for each axis based on the average size of the bounding box sides of M_{ref} (line 1). Next, the centroid c_{ref} is computed from M_{ref} (line 2). The rotation matrix of the medical data R_{med} is initially set to be a pre-defined initial pose R_0 (line 3). Pose R_{roi} and center of mass c_{roi} of patient's region of interest are estimated with a trained pose estimator based on the region of interest (lines 4-5). c_{roi} is assigned to be the translation vector of the region of interest: t_{roi} (line 6). As the region of interest in this work is a head, we estimate these values from the current depth map captured by the Kinect sensor with the real-time head pose estimation proposed in [26]. The centroid of the region of interest's center of mass rotated c_{rot} is computed (line 7). Because we want the centroids of the medical data and the reference model to be the same (i.e. at the same position), and assuming that the initial centroid of the volume is in the origin, the translation vector of the medical data can be computed by the subtraction of c_{ref} and c_{rot} (line 8).

Algorithm 1 Estimating coarse pose

1: $S_{med} \leftarrow$ compute scale factor from M_{ref}.
2: $c_{ref} \leftarrow$ compute centroid from M_{ref}.
3: $R_{med} \leftarrow R_0$.
4: $R_{roi} \leftarrow$ estimate rotation matrix for the region of interest.
5: $c_{roi} \leftarrow$ estimate centroid of the region of interest.
6: $t_{roi} \leftarrow c_{roi}$.
7: $c_{rot} \leftarrow R_{roi} c_{roi}$.
8: $t_{med} \leftarrow c_{ref} - c_{rot}$.

After the computation of the coarse pose, the medical volume data V can be positioned into the scene for every frame by the following expression:

$$V_{live} = T_{live} T_{med} T_{roi} V_{base}. \quad (1)$$

[1] Point cloud.

[2] In this paper, we refer to normal map as an array which stores the normal vector for each vertex in the vertex map.

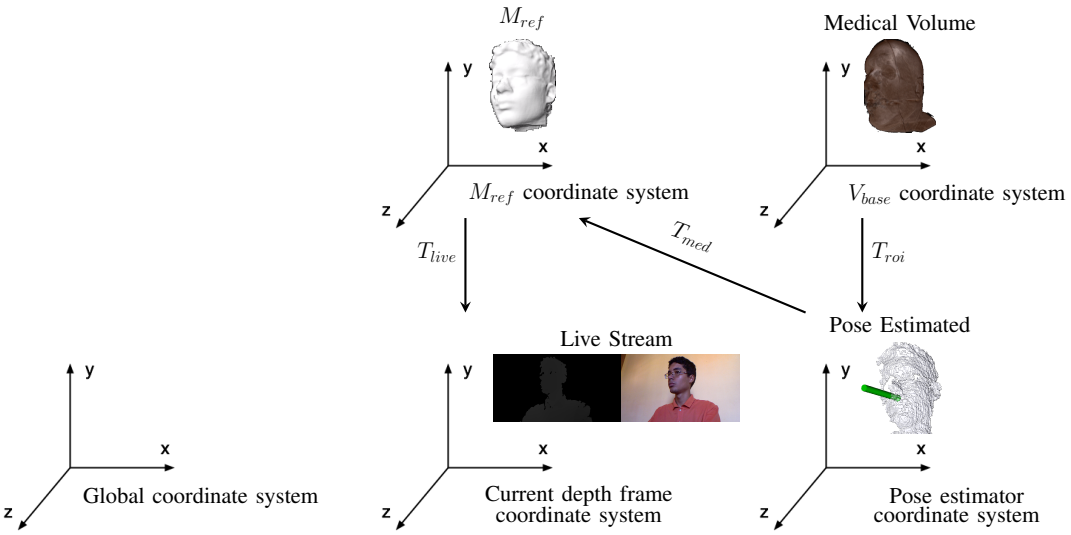

Fig. 2. Relationships among the coordinate systems used for medical volume-to-patient registration. 3D cartesian coordinate system courtesy of VRArchitect[2]

where T_{live} is the 3D rigid transformation estimated by the tracking algorithm. $T_{roi} = [R_{roi}|t_{roi}]$ is the 3D rigid transformation that registers the medical data in relation to the pose measured for the region of interest, V_{base} means the volume in an initial position, and T_{med}, which is defined by the following equation:

$$T_{med} = [R_{med} diag(S_{med})|t_{med}]. \quad (2)$$

is the 3D transformation that gives the coarse placement of the volume into the scene. The function $diag$ creates a 3×3 diagonal matrix from the input vector. The final result, V_{live}, is the volume aligned to the patient in the current frame.

Once in the origin on the unitary bounding box, the sequence of transformations can be applied to place the volume on the current pose of the patient's region of interest. The user can adjust volume's position and orientation by controlling T_{med}. The next subsection explains how T_{live} is estimated.

E. Live Tracking

Live tracking is done in two steps: during reconstruction of the 3D reference model and during MAR with the patient and the medical data. Instead of using an image-based tracking, we use a real-time variant of the Iterative Closest Point (ICP) [27], [28]. As done in [29], we choose a depth-based method because it does not suffer from changes in illumination or presence of complex texture patterns in the scene. The ICP is used to estimate the transformation that aligns the current depth frame captured by the RGB-D sensor with the previous one represented by the 3D reference model.

In presence of fast rotations and translations in the scene, the real-time variant of the ICP algorithm may fail. To minimize this problem, a real-time head pose estimation algorithm is used to give a new initial guess to the ICP to correctly compute the current transformation [30]. The head pose estimation used is the algorithm proposed by Fanelli et al. [26]. This probabilistic approach achieves high accuracy and runs in real-time even in a single CPU.

After the computation of the rigid transformation, it is applied to the medical data, which can be composed with the real scene captured by the Kinect.

IV. On-Patient Volumetric Medical Data Visualization

In this section we describe the techniques used to render the medical volume (Figure 1-middle layer) and to integrate the volume rendered into the MAR environment (Figure 1-bottom layer).

A. Volume Rendering

Volume rendering is concerned with techniques for generating images from volume data [8]. The majority of volume rendering algorithms are based on the volume rendering integral. This formulation is based on a emission-absorption optical model as shown in Equation 3.

$$I(D) = I_0 e^{-\int_{s0}^{D} k(t)dt} + \int_{s0}^{D} q(s) e^{-\int_{s}^{D} k(t)dt} ds. \quad (3)$$

The radiance energy *I(D)* is the result of integrating from entry point into the volume ($s = s_0$) to the exit point toward the camera ($s = D$). The absorbed energy and emission components are represented by the absorption and emission coefficients k and q respectively. The term I_0 is the radiance in the entry point s_0.

The volume is rendered according to a compositing scheme, which gives the numerical computation of the volume rendering integral:

$$I(D) = \sum_{i=0}^{n} c_i \prod_{j=i+1}^{n} T_j \quad (4)$$

[2]http://www.vrarchitect.net/anu/cg/Maths/cartesian3D.en.html

where $c_i = I(s_i)$ and $T_i = T(s_{i-1}, s_i) = e^{-\int_{s_{i-1}}^{s_i} k(t)dt}$.

Two of the most known compositing schemes are the direct volume rendering (DVR) and the maximum intensity projection (MIP). The DVR is the discretization presented in the Equation 4 and it is based on a front-to-back or back-to-front compositing. The most common is the front-to-back DVR:

$$C_{dst} = C_{dst} + (1 - \sigma_{dst})C_{src} \tag{5}$$

$$\sigma_{dst} = \sigma_{dst} + (1 - \sigma_{dst})\sigma_{src} \tag{6}$$

where $C_{dst} = c_{i+1}$, $C_{src} = c_i$, $\sigma_{dst} = 1 - T_{i+1}$, $\sigma_{src} = 1 - T_i$ given the voxel i being traversed. C represents the color contribution and σ the opacity of the voxel.

Different from the DVR compositing scheme, MIP is computed according to the following compositing equation:

$$C_{dst} = max(C_{dst}, C_{src}) \tag{7}$$

The final result is the maximum color contribution along a ray [8]. This compositing scheme is particularly important in the virtual angiography (i.e. the display of the vessel structures in medical scans) [31].

The volume data is represented as a 3D texture with associated colors. This representation allows the generation of images with higher quality than the 2D texture-based solution [8]. To render the medical data based on DVR or MIP compositing scheme, the ray casting technique is used. The start positions of the ray are obtained by rasterizing the front faces of the volume bounding box and the exit positions of the ray are obtained by rasterizing the back faces of the bounding box. Direction is computed from the difference between the exit and start positions. Ray casting is performed by sampling the space in-between the volume bounding box by using an adaptive sampling rate (which is discussed below). Ray casting is done on GPU in a single rendering pass on the fragment shader [32].

One of the main advantages of ray casting is that it is flexible in the sense that many other techniques can be integrated to improve image quality or performance of the rendering.

To reduce sampling artifacts, a stochastic jittering (i.e. random ray-start off-setting) is applied to the ray start position. To reduce the filtering artifacts, a fast GPU-Based tri-cubic filtering [33], [34] and a GPU pre-filter for accurate tri-cubic filtering [35] are used.

The performance of our volume rendering is optimized by empty-space leaping the non-visible voxels. The volume is subdivided into an octree. In order to detect empty space, each block stores the minimum and maximum scalar values. The visibility of each block can be determined after evaluation of the transfer function [36]. If the block is considered invisible, the step size of the ray is increased, otherwise, it is decreased. Our approach also supports early ray termination, if the opacity accumulated is greater than a threshold, and image downscaling, when the volume size is not supported by the graphics rendering.

The volume data consists in scalar values that represent some spatially varying physical property. Transfer functions can be applied on these scalar values to improve the user's visual perception and data interpretation of the volume. The transfer functions map the values to colors in the RGB space. In this work, pre-integrated transfer functions [37] are used to capture the high frequencies introduced in the transfer functions with low sampling rates.

The volume rendering integral presented in the Equation 3 does not account for illumination effects caused by external light sources. Such illumination effects, however, add a great deal of realism to the resulting images. This is specially important in an AR environment, where this illumination effect serves as an approximation of the illumination of the real scene. To compute local illumination, it is used Blinn-Phong shading [38] with on-the-fly gradient computation by central or forward differences on the GPU. Non-polygonal iso-surface rendering is realized by first hit ray-casting. The local illumination is included in the Equation 3 by extending the emission coefficient $q(s) = q_{ea}(s) + q_{il}(s)$, where $q_{ea}(s)$ is the emission coefficient of the emission-absorption model and $q_{il}(s)$ is the coefficient that adds the local illumination [8].

B. Context-Preserving Volume Rendering

When the medical volume is rendered based on DVR, at first glance, the final rendering may not be the result desired by the user. In this sense, transfer functions can be used to change the visual aspect of the volume, enhancing the features of the medical data. However, the process to find an appropriate transfer function can be a complex task and time-consuming. An alternative is to pre-segment the volume in regions of interest and define transfer functions for these regions separately. This solution improves the quality of the final rendering, but it is more time-consuming than defining the transfer function without pre-segmentation.

Volume clipping can be used to reveal hidden structures of the volume by completely cutting away occluding areas. Traditionally, clipping planes are used to perform this task in a faster and intuitive way [8]. However, when clipping a volume, some important information can be lost in this process if the information is located in the cutted region. Therefore, volume clipping is not the best way to enhance the visualization of the volume.

Illustrative volume rendering is a field which aims to improve the visualization of volumetric structures by the use of non-photorealistic strategies integrated into the rendering algorithm. Inspired by the fields of volume clipping and illustrative volume rendering, Bruckner et al. proposed the context-preserving volume rendering [4], [5], which uses a function of:

1) Shading intensity - to decrease opacity in regions which receive high illumination intensity;
2) Gradient magnitude - to preserve and enhance contours;
3) Distance to the eye point - to simulate the cut effect of a clipping plane;
4) Previously accumulated opacity - to turn structures located behind semi-transparent regions more opaque.

Fig. 3. Some of the visualization options. A) Direct volume rendering (DVR). B) DVR with pre-integrated transfer function. C) DVR with pre-integrated transfer function and Blinn-Phong illumination. D) Non polygonal iso surface volume rendering.

This model is able to reduce the opacity in less important data regions and it is controlled by two user-specified parameters: one to explore the dataset and another to control the sharpness of the visualization.

In our environment, we use this technique not only to enhance features of the volume, but also to separate some structures in the visualization, such as bones, soft tissue and organs (see Figure 1-middle layer). This process can be done just by controlling the two parameters mentioned before.

C. Focus + Context Visualization

An application for on-patient medical data visualization requires a special attention to the mixing between real (i.e. patient's image captured by the sensor) and virtual (i.e. patient's anatomy stored in the computer) entities of the AR environment.

Bichlmeier et al. proposed the Contextual Anatomic Mimesis to improve depth perception in a medical AR application [2]. First, the medical data is not entirely overlaid over the patient's image. Focus point and radius are defined by line of sight. Together, they act like a mask in the AR scene, in which the medical data can be visualized only inside it. Next, Bichlmeier and colleagues proposed the control over the AR visualization by adjusting parameters such as:

1) Curvature (α_{curv}) - The curvature of the patient's skin surface allows regions with high curvature (e.g. wrinkled, bumpy and sinuous regions) to remain visible in the final rendering;
2) Distance Falloff ($\alpha_{distFalloff}$) - The distance between each point on the surface and the focus point allows a smooth visualization between the patient's medical data and the real image;

This method was integrated in our environment, following the pipeline of the bottom layer of Figure 1, because it has already proven to provide high-quality on-patient medical data visualization [7], [9].

D. Integration into a MAR environment

According to Figure 1, after the volume rendering we read the color frame buffer of the volume and send it to a shader to blend it with the RGB data coming from the Kinect sensor. The blending is done by the following linear interpolation:

$$I_{final} = \beta I_{real} + (1 - \beta) I_{medical} \quad (8)$$

where I_{real} is the image captured by the sensor, $I_{medical}$ is the image corresponding to the medical volume, and I_{final} is the resulting image. In our approach, β is defined dynamically, for every fragment/pixel, by the F+C visualization technique described in the previous subsection, according to the following equation:

$$\beta = clamp(max(\alpha_{curv}, \alpha_{distFalloff})) \quad (9)$$

$clamp$ is a function that clamps its input parameter to the interval [0,1].

Incorrect occlusion of virtual and real objects in an augmented scene is one of the fundamental issues in AR applications. To solve it, the depth maps of the 3D reference object reconstructed previously and the 3D object coming from the sensor's live stream are used. If the live object is in front of the reference object, the volume is the occludee, otherwise, it is the occluder.

Blending and occlusion are computed in a GLSL (OpenGL Shading Language) fragment shader that process the color and depth buffers, respectively, to do these operations.

V. Results and Discussion

In this section we analyze visual quality and performance of the whole approach and particularly the techniques employed for volume rendering.

A. Experimental Setups

For all tests we used an Intel® Core™ i7-3770K CPU@3.50Ghz, 8GB RAM, NVIDIA GeForce GTX 660. We used the open-source C++ implementation of the KinectFusion [23] released by PCL project [39].

The medical datasets used are a magnetic resonance (MR) volume of a head[3] of two different resolutions: 256^3 and 512^3, and a CT volume of a head from the Visible Human Project[4] of resolution $128 \times 256 \times 256$. The reference human face was reconstructed with the KinectFusion using a grid with volume size of 70cm $\times$ 70cm $\times$ 140cm and resolutions of 256^3 and 512^3.

We evaluate our approach in a scenario where the patient's head is augmented with a generic volume dataset of a head. The use of a generic volume does not affect our performance

[3]http://graphics.stanford.edu/data/voldata/
[4]http://www.nlm.nih.gov/research/visible/

evaluation, as it is of a typical volume size. Also, the generic volume does not affect our visual quality evaluation since the volume is semi-automatically adjusted on patient's head.

B. Performance Evaluation

We have evaluated the performance of the proposed approach in four tests. In the first test, the time required for each step of our approach was measured in an application with the resolution of KinectFusion's grid and medical data given by 512^3 and 256^3, respectively. The goal of this test is to evaluate the performance for each step individually.

Figure 4 shows the time measured for each step of 3D reference model reconstruction. In summary, it takes $\approx$ 25ms per frame (40 FPS) and requires about 15 seconds to be completed if the user has provided sufficient viewpoints. This processing time could be improved by reducing KinectFusion's volume resolution from 512^3 to 256^3, which would reduce 3D reference model accuracy as well. Based on the processing times shown in Figure 5, by using the resolution of 256^3, 3D reference model reconstruction takes $\approx$ 19ms per frame (53 FPS). All the performance tests performed in this subsection were done by using maximum resolution of 512^3. From the Figure 5, one can compute the corresponding processing time by using the resolution 256^3 for the other evaluations.

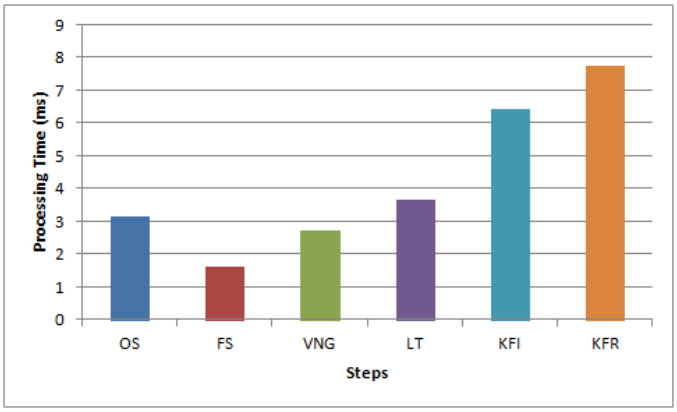

Fig. 4. Performance results measured in average milliseconds for each step of our approach. OS - Other Steps (i.e. display timing, Kinect latency), FS - Face Segmentation, VNG - Vertex and Normal Map Generation, LT - Live Tracking, KFI - KinectFusion's grid integration, KFR - KinectFusion's grid raycasting. Times were measured running our approach with the KinectFusion's grid in resolution 512^3.

Figure 6 shows the time measured for each step of the MAR live tracking. It takes $\approx$ 22ms per frame (45 FPS). Occlusion computation, which transfers data stored in GPU to CPU, converts 3D reference model and 3D object coming from Kinect to the same coordinate system and sends their depth maps to the shader, takes 5ms in our approach. Meanwhile, our optimized direct volume rendering takes the lowest time.

As the Kinect sensor provides depth and color maps at 30 FPS, our approach can process every input frame during 3D reference model reconstruction and MAR live tracking with on-patient medical data visualization. Therefore, we can conclude that our approach runs in real-time.

The solution proposed in [30] was not included in the measurement of the live tracking because it does not run for every

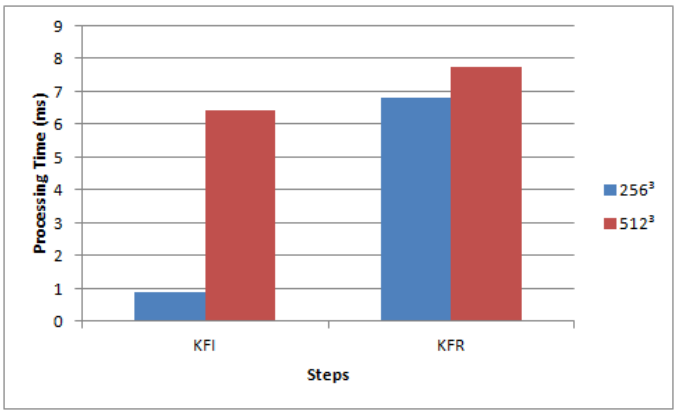

Fig. 5. Performance results measured in average milliseconds for each step of the KinectFusion. KFI - KinectFusion's grid integration. KFR - KinectFusion's grid raycasting. Times were measured running our approach with KinectFusion's grid in resolutions 256^3 and 512^3.

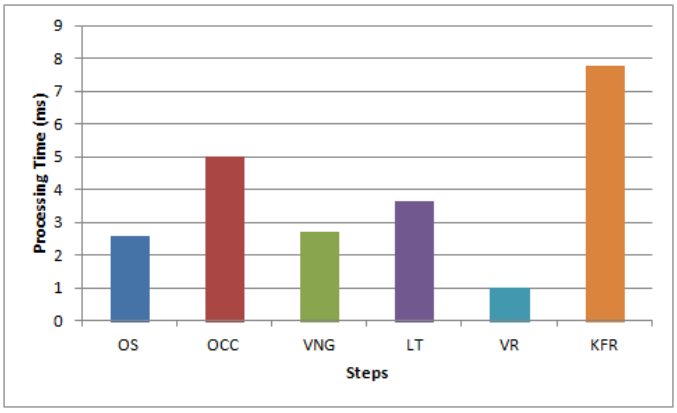

Fig. 6. Performance results measured in average milliseconds for each step of our approach. OS - Other Steps (i.e. display timing, Kinect latency), OCC - Occlusion Computation, VNG - Vertex and Normal Map Generation, LT - Live Tracking, VR - Volume Rendering, KFR - KinectFusion's grid raycasting. Times were measured running our approach with the KinectFusion's grid in resolution 512^3 and medical dataset in resolution 256^3.

input frame. When used, it added 40ms in total frame time. We have observed that the user takes about 10 seconds to position and adjust the volume in the scene. The algorithm for coarse medical volume-to-patient registration is used only once (i.e. at the transition between 3D reference model reconstruction and on-patient medical data visualization) and takes 60ms. Focus+context visualization runs at maximum application's performance as it operates directly on the shader.

In the second test, we evaluate the influence of the volume resolution on the overall performance of our approach. It has never dropped below 29 FPS using medical and KinectFusion volumes of resolution 256^3 and 512^3 with DVR. Therefore, we can use the maximum KinectFusion's volume size to generate a more accurate 3D reference model.

In the third test, the average processing time for various volume rendering compositing schemes was measured. The performance results can be seen in Figure 7. From the analysis on the first test, if the volume rendering takes less than

10ms, the application still keeps the real-time performance. By assuming that the typical resolution of a head medical volume is 256^3[5] and considering the reported processing times in Figure 7, we conclude that our approach runs in real-time because its performance is greater than 30 FPS. However, with a volume of resolution 512^3, depending on which mode is chosen, we have a loss in the performance of the application.

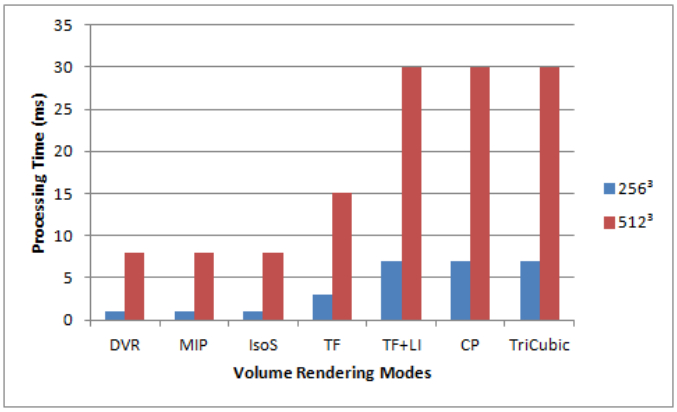

Fig. 7. Performance results measured in average milliseconds for various volume rendering compositing schemes. DVR - Direct Volume Rendering. MIP - Maximum Intensity Projection. IsoS - Non-Polygonal Iso Surface. TF - DVR + Transfer Function with Pre-Integration. LI - DVR + Local Illumination via Blinn-Phong shading. CP - Context-Preserving Volume Rendering. TriCubic - Fast Tricubic Filtering.

As described in Section IV-A, the volume is stored as a discrete 3D texture. When the ray is casted into the volume, it accesses the space between the discrete samples of the volumetric data. In this case, trilinear interpolation is used to reconstruct a continuous representation of the volume based on the eight closest neighbours samples of that space. This is the most expensive operation in the volume rendering based on ray casting as it requires eight memory access to perform the interpolation. Based on this statement, it is possible to evaluate the variation in performance of the different volume rendering modes.

In the simplest DVR, trilinear interpolation is performed only once for each position of the ray casted. Therefore, this is the rendering mode which takes the lowest processing time. As consequence, it produces the simplest visual effects, which can be seen in Figure 3-A.

For the MIP, a variation of DVR mode, the trilinear interpolation is also done only once. Therefore, it has the same performance measured for DVR mode.

In non-polygonal iso surface rendering, Blinn-Phong shading is computed when the ray traverses a voxel with isovalue greater than a threshold defined semi-automatically. The normal vector for a given voxel is computed by the normalization of the central differences of the neighbouring voxels. This gradient estimation requires six trilinear interpolations. However, as it is not computed for every voxel being traversed, it does not increase significantly the computational cost of the volume rendering. An example of non-polygonal iso surface rendering can be seen in Figure 3-D.

[5]http://www9. informatik.uni-erlangen.de/External/vollib/

In the DVR with pre-integrated transfer function, after the trilinear interpolation of the voxel, the scalar value of the previous and current voxel being traversed are used as a look-up in a 2-D pre-integration table. This lookup is performed with a bilinear interpolation. It increases the volume rendering processing time to 15 ms per frame and slightly decreases the performance of the application to 35 ms per frame (28 FPS). Despite of this fact, we have a more pleasant visualization of the volume. An example of such effect can be seen in Figure 3-B.

In the DVR with transfer function and local illumination, for every voxel being traversed, the transfer function is accessed (with a bilinear interpolation) and the illumination is computed (with six trilinear interpolations). These interpolations decrease significantly the performance of the volume rendering to 30 ms per frame and the application to 50 ms per frame (20 FPS), which is not prohibitive, as the user can still interact with the application with some delay. In the final result, the illumination effects add realism to the resulting image. An example can be seen in Figure 3-C.

Context-Preserving volume rendering adds some computation for DVR with transfer function and local illumination. However, there is not a new trilinear interpolation to be performed. Therefore, Context-Preserving mode has the same performance as the case mentioned above.

Fig. 8. A volume rendering (left) with stochastic jittering (center) and tri-cubic filtering (right). The stochastic jittering reduces the wood-grain artifacts in the volume, however it is almost imperceptible in this scene. The tri-cubic filtering smoothes the volume data, reducing the artifacts present in the volume rendered with trilinear filtering.

In the test performed with a simple DVR and fast tri-cubic filtering, for every voxel being traversed, eight trilinear interpolations are computed to return one tricubic interpolation. These interpolations decrease the performance similarly to the situation of transfer function and local illumination. The volume requires 30 ms per frame to be rendered and the application requires 50 ms per frame (20 FPS). The visual influence of the tri-cubic interpolation against the trilinear one can be seen in Figure 8.

C. Visual Quality Evaluation

As mentioned in Section IV-A, the proposed approach supports various volume rendering modes. Some of them can be seen in Figure 3. As discussed in previous subsection, each one of them has some impact on application's performance, although they still run in real-time for typical-sized medical data.

Figure 8 shows the influence of some techniques used to improve the image quality of the volume rendering. Artifacts

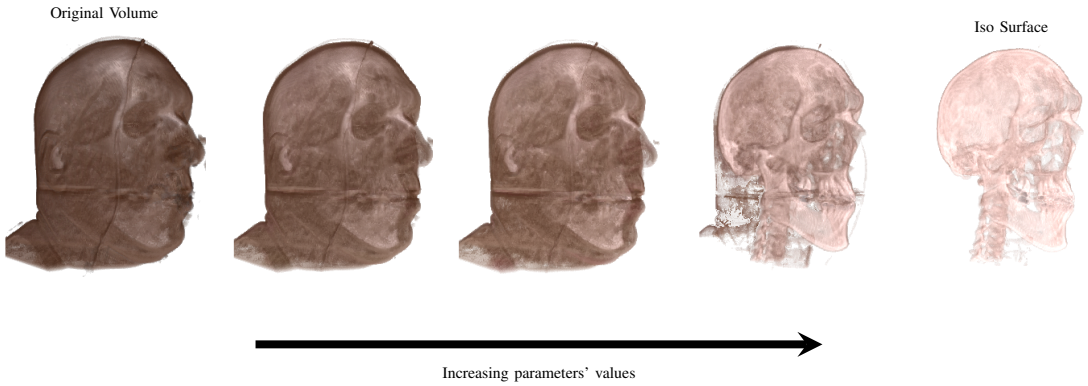

Fig. 9. From volume to iso surface rendering by controlling context-preserving volume rendering parameters.

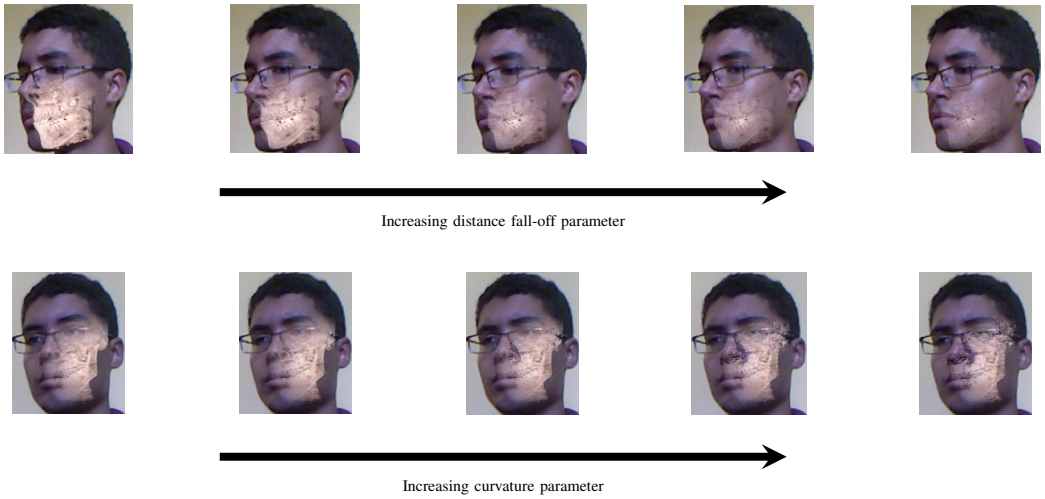

Fig. 10. Focus+Context visualization to improve human's perception of the augmented scene.

are reduced without prohibitive increase in the computational cost.

As can be seen in Figure 1, our approach supports occlusion at shader level. However, if the occluder overlaps more than 70% of the occludee, tracking may fail.

Influence of context-preserving volume rendering parameters in our application can be seen in Figure 9. By increasing the values of the two parameters proposed by Bruckner et al. [4], [5], external structures of the volume (e.g. soft tissue) become increasingly invisible, arising the visualization of internal structures of the volume (e.g. bone). Support for this kind of rendering is specially important in our application, as a physician does not necessarily want to visualize the naive volume rendering, with soft tissue, organs and bones altogether.

Influence of focus+context visualization parameters proposed by Bichlmeier et al. [2] in our MAR environment can be seen in Figure 10. As the distance fall-off increases, transition between real and virtual images becomes smoother and volume less visible. With respect to the curvature term, as it increases, regions of the real scene with high curvature (i.e. nose and glasses in Figure 10) remain visible even if they are inside the focus region.

VI. Conclusion and Future Work

On-patient medical data visualization can be used to improve medical diagnosis, surgical planning, training, operation and post-operative examination. In this paper, we have presented a marker-free augmented reality approach for on-patient volumetric medical data visualization. We used the KinectFusion algorithm to reconstruct the patient's head and a variant of the ICP algorithm in conjunction with a face tracking solution to track it during the MAR. We have tested and applied standard volume rendering techniques to render volumes with high quality and shown that, with a typical volume size, the proposed algorithm is capable to run in real-time and it provides high visual quality for the final augmented scene through the use of focus + context visualization. Moreover, it provides accuracy enough for applications that need good "visual" accuracy for the registration (i.e. good composition and tracking of the virtual object into the augmented scene). Experiments must still be conducted to evaluate if the accuracy achieved is enough for medical AR applications to aid surgery operation, for instance.

Markerless tracking fails if patient's ROI is not visible in the scene. Relocalization techniques can be used to solve this problem. For the MAR environment, we have used a conventional display to show the augmented scene. Multi-view solutions based on AR glasses or portable solutions based on mobile devices can be employed by processing the proposed approach on a server and transferring the visualization of the augmented content for these alternative hardwares, allowing a more seamlessly visualization of the virtual content onto the real scene. Also, inspired by the field of image-based lighting, real local illumination and fast global illumination could be applied to improve the realism of the volume rendering and the integration with the real scene.

ACKNOWLEDGMENTS

We are grateful to the PCL project for providing the open-source implementation of the KinectFusion algorithm. This research is financially supported by FAPESB and CAPES.

REFERENCES

[1] M. Macedo, A. Apolinario, A. C. Souza, and G. A. Giraldi, "A Semi-Automatic Markerless Augmented Reality Approach for On-Patient Volumetric Medical Data Visualization," in *SVR*, Brazil, 2014.

[2] C. Bichlmeier, F. Wimmer, S. M. Heining, and N. Navab, "Contextual anatomic mimesis hybrid in-situ visualization method for improving multi-sensory depth perception in medical augmented reality," ser. ISMAR '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 1–10.

[3] J. Mercier-Ganady, F. Lotte, E. Loup-escande, M. Marchal, and A. Lecuyer, "The mind-mirror: See your brain in action in your head using eeg and augmented reality," in *Virtual Reality (VR), 2014 iEEE*, March 2014, pp. 33–38.

[4] S. Bruckner, S. Grimm, A. Kanitsar, and M. E. Gröller, "Illustrative context-preserving volume rendering," in *Proceedings of the Seventh Joint Eurographics / IEEE VGTC conference on Visualization*, ser. EUROVIS'05. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2005, pp. 69–76.

[5] S. Bruckner, S. Grimm, A. Kanitsar, and M. Groller, "Illustrative context-preserving exploration of volume data," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 12, no. 6, pp. 1559–1569, Nov 2006.

[6] S. K. Card, J. D. Mackinlay, and B. Shneiderman, Eds., *Readings in Information Visualization: Using Vision to Think*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999.

[7] O. Kutter, A. Aichert, C. Bichlmeier, J. Traub, S. M. Heining, B. Ockert, E. Euler, and N. Navab, "Real-time Volume Rendering for High Quality Visualization in Augmented Reality," in *International Workshop on Augmented environments for Medical Imaging including Augmented Reality in Computer-aided Surgery (AMI-ARCS 2008)*. New York, USA: MICCAI Society, Sept. 2008.

[8] M. Hadwiger, J. M. Kniss, C. Rezk-salama, D. Weiskopf, and K. Engel, *Real-time Volume Graphics*. Natick, MA, USA: A. K. Peters, Ltd., 2006.

[9] M. Wieczorek, A. Aichert, O. Kutter, C. Bichlmeier, J. Landes, S. M. Heining, E. Euler, and N. Navab, "GPU-accelerated Rendering for Medical Augmented Reality in Minimally-Invasive Procedures," in *Proceedings of BVM 2010*. Springer, Mar. 2010.

[10] H. G. Debarba, J. Grandi, A. Maciel, and D. Zanchet, "Anatomic hepatectomy planning through mobile display visualization and interaction." in *MMVR*, ser. Studies in Health Technology and Informatics, vol. 173. IOS Press, 2012, pp. 111–115.

[11] H. Suenaga, H. Hoang Tran, H. Liao, K. Masamune, T. Dohi, K. Hoshi, Y. Mori, and T. Takato, "Real-time in situ three-dimensional integral videography and surgical navigation using augmented reality: a pilot study," *International Journal of Oral Science*, no. 2, p. 98–102, 2013.

[12] L. Maier-Hein, A. M. Franz, M. Fangerau, M. Schmidt, A. Seitel, S. Mersmann, T. Kilgus, A. Groch, K. Yung, T. R. dos Santos, and H.-P. Meinzer, "Towards mobile augmented reality for on-patient visualization of medical images." in *Bildverarbeitung für die Medizin*, ser. Informatik Aktuell, H. Handels, J. Ehrhardt, T. M. Deserno, H.-P. Meinzer, and T. Tolxdorff, Eds. Springer, 2011, pp. 389–393.

[13] T. R. dos Santos, A. Seitel, H.-P. Meinzer, and L. Maier-Hein, "Correspondences search for surface-based intra-operative registration," ser. MICCAI'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 660–667.

[14] L. Maier-Hein, M. Schmidt, A. M. Franz, T. R. dos Santos, A. Seitel, B. Jähne, J. M. Fitzpatrick, and H. P. Meinzer, "Accounting for anisotropic noise in fine registration of time-of-flight range data with high-resolution surface data," in *Proceedings of the 13th international conference on Medical image computing and computer-assisted intervention: Part I*, ser. MICCAI'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 251–258.

[15] J.-D. Lee, C.-H. Huang, T.-C. Huang, H.-Y. Hsieh, and S.-T. Lee, "Medical augment reality using a markerless registration framework." *Expert Syst. Appl.*, vol. 39, no. 5, pp. 5286–5294, 2012.

[16] T. Blum, V. Kleeberger, C. Bichlmeier, and N. Navab, "mirracle: Augmented reality in-situ visualization of human anatomy using a magic mirror," in *Virtual Reality Short Papers and Posters (VRW), 2012 IEEE*, 2012, pp. 169–170.

[17] M. Meng, P. Fallavollita, T. Blum, U. Eck, C. Sandor, S. Weidert, J. Waschke, and N. Navab, "Kinect for interactive ar anatomy learning," in *ISMAR*, Adelaide, Australia, October 2013.

[18] P. Stefan, P. Wucherer, Y. Oyamada, M. Ma, A. Schoch, M. Kanegae, N. Shimizu, T. Kodera, S. Cahier, M. Weigl, M. Sugimoto, P. Fallavollita, H. Saito, and N. Navab, "An ar edutainment system supporting bone anatomy learning," in *Virtual Reality (VR), 2014 iEEE*, March 2014, pp. 113–114.

[19] L. Cruz, D. Lucio, and L. Velho, "Kinect and rgbd images: Challenges and applications," in *Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2012 25th SIBGRAPI Conference on*, 2012, pp. 36–49.

[20] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vision*, vol. 57, no. 2, pp. 137–154, May 2004.

[21] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, 1998. Sixth International Conference on*, jan 1998, pp. 839 –846.

[22] S. Holzer, R. Rusu, M. Dixon, S. Gedikli, and N. Navab, "Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 2684–2689.

[23] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, ser. UIST '11. New York, NY, USA: ACM, 2011, pp. 559–568.

[24] S. J. Osher and R. P. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, 2003rd ed. Springer, Oct. 2002.

[25] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '96. New York, NY, USA: ACM, 1996, pp. 303–312.

[26] G. Fanelli, T. Weise, J. Gall, and L. V. Gool, "Real time head pose estimation from consumer depth cameras," in *33rd Annual Symposium of the German Association for Pattern Recognition (DAGM'11)*, September 2011.

[27] P. Besl and H. McKay, "A method for registration of 3-d shapes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 2, pp. 239 –256, feb 1992.

[28] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vision Computing*, vol. 10, no. 3, pp. 145 – 155, 1992.

[29] M. Macedo, A. L. Apolinario, and A. C. Souza, "A Markerless Augmented Reality Approach Based on Real-Time 3D Reconstruction using Kinect," in *Workshop of Works in Progress (WIP) in SIBGRAPI 2013 (XXVI Conference on Graphics, Patterns and Images)*, S. M. Alejandro C. Frery, Ed., Arequipa, Peru, august 2013.

[30] M. Macedo, A. Apolinario, and A. C. Souza, "A Robust Real-Time Face Tracking using Head Pose Estimation for a Markerless AR System," in *SVR*, Cuiaba, MT, Brazil, May 2013.

[31] M. Prokop, H. O. Shion, A. Schanz, and M. Andschaefer-Prokop, "Use of maximum intensity projections in ct angiography," in *Radiographics*, vol. 17, 1997, pp. 433–451.

[32] M. Hadwiger, P. Ljung, C. R. Salama, and T. Ropinski, "Advanced illumination techniques for gpu-based volume raycasting," in *ACM SIGGRAPH 2009 Courses*, ser. SIGGRAPH '09. New York, NY, USA: ACM, 2009, pp. 2:1–2:166.

[33] C. Sigg and M. Hadwiger, "Fast third-order texture filtering," in *GPU Gems 2*, M. Pharr, Ed. Addison-Wesley, 2005, pp. 313–329.

[34] D. Ruijters, B. M. ter Haar Romeny, and P. Suetens, "Efficient gpu-based texture interpolation using uniform b-splines." *J. Graphics Tools*, vol. 13, no. 4, pp. 61–69, 2008.

[35] D. Ruijters and P. Thévenaz, "Gpu prefilter for accurate cubic b-spline interpolation." *Comput. J.*, vol. 55, no. 1, pp. 15–20, 2012.

[36] W. Li, K. Mueller, and A. Kaufman, "Empty space skipping and occlusion clipping for texture-based volume rendering," in *Visualization, 2003. VIS 2003. IEEE*, 2003, pp. 317–324.

[37] K. Engel, M. Kraus, and T. Ertl, "High-quality pre-integrated volume rendering using hardware-accelerated pixel shading," in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, ser. HWWS '01. New York, NY, USA: ACM, 2001, pp. 9–16.

[38] J. F. Blinn, "Models of light reflection for computer synthesized pictures," in *Proceedings of the 4th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '77. New York, NY, USA: ACM, 1977, pp. 192–198.

[39] R. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 1 –4.