Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: www.elsevier.com/locate/cmpb

Corrigendum

ARmedViewer, an augmented-reality-based fast 3D reslicer for medical image data on mobile devices: A feasibility study

Bragi Sveinsson[a,b,*], Neha Koonjoo[a,b], Matthew S Rosen[a,b,c]

[a] A.A. Martinos Center for Biomedical Imaging, Massachusetts General Hospital, Boston, MA, United States
[b] Harvard Medical School, Boston, MA, United States of America
[c] Department of Physics, Harvard University, Cambridge, United States of America

ARTICLE INFO

Article history:
Received 30 June 2020
Accepted 8 November 2020

Keywords:
Augmented reality
Medical imaging
Mobile device

ABSTRACT

Background and objective: Medical images obtained by methods such as magnetic resonance imaging (MRI) or computed tomography (CT) are typically displayed as a stack of 2D slices, comprising a 3D volume. Often, the anatomy of interest does not fall neatly into the slice plane but rather extends obliquely through several slices. Reformatting the data to show the anatomy in one slice in conventional medical imaging software can require expertise and time. In this work, we present ARmedViewer, a medical image viewing app designed for mobile devices that uses augmented reality technology to display medical image data. An arbitrary plane for displaying the data can be chosen quickly and intuitively by moving the mobile device.

Methods: The app ARmedViewer, compiled for an iOS device, was designed to allow a user to easily select from a list of 3D image datasets consisting of header information and image data. The user decides where to place the data, which can be overlaid on actual human anatomy. After loading the dataset, the user can move and rotate the data as desired. 15 users compared the user experience of the app to a common image viewer by answering two user surveys each, one custom and one standardized. The utility of the app was also tested by having two users find a plane through a 3D dataset that displayed 3 randomly placed lesions. This operation was timed and compared between the app and a standard medical image viewer.

Results: ARmedViewer was successfully developed and run on an iPhone XS. User interfaces for selecting, placing, moving, reslicing, and displaying the data were operated with ease, even by naïve users. The custom user survey indicated that freely selecting a slice through the data was significantly more intuitive and easier using the app than using a conventional image viewer on a computer workstation, and changing the viewing angle was also significantly more intuitive. The standardized survey indicated a significantly better user experience for the app in several categories, and never significantly worse. The timed reslicing experiments demonstrated the app being faster than the standard image viewer by an average factor of 9.

Conclusions: The newly developed ARmedViewer is a portable software tool for easily displaying 3D medical image data overlaid on human anatomy, allowing for easy choice of the viewing plane by intuitively moving the mobile device.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

Modern medical imaging can be performed with multiple imaging techniques, including magnetic resonance imaging (MRI), computed tomography (CT), and positron emission tomography (PET). Typically, such tomographic image data is displayed as a series of slices that are combined to form a volume, using medical image data formats such as DICOM [1] or NIfTI [2]. Using medical image viewing software on a computer workstation, an end user can scroll through the slices to view anatomical details [3–5]. For instance, after a patient has been scanned in a clinical MRI scanner located at a hospital or similar medical institution, the images are read by radiologists in a specially designed reading room

* Corresponding author: Bragi Sveinsson, 149 13th Street, Suite 2301, Boston, MA 02129.
E-mail address: bsveinsson@mgh.harvard.edu (B. Sveinsson).

https://doi.org/10.1016/j.cmpb.2020.105836
0169-2607/© 2020 Elsevier B.V. All rights reserved.

with powerful workstations and professional medical image viewing software, such as OsiriX or Horos [6,7].

Recently, much progress has been made in the development of more portable, less expensive medical imaging devices [8,9]. This includes magnetic resonance imaging with electromagnets [10,11] and permanent magnets [12–16] for mobile point-of-care imaging as well as more low-cost imaging in underserved areas such as rural regions or developing countries. However, the medical image viewing paradigm has remained mostly the same, i.e., a radiologist reads the images on a computer workstation in a dedicated reading room.

Furthermore, using medical image viewing software to display data in the desired manner can be non-trivial, requiring substantial expertise. For example, the anatomy of interest may not always align well with the acquired image plane [17]. In some cases it has been shown to be beneficial to view the data in multiple non-perpendicular scan planes [18]. While reformatting the slices to display a plane orthogonal to the original acquired slice plane is often straightforward, a completely free choice of an oblique viewing plane can be a more laborious process [19]. Additionally, overlaying the images on the actual anatomy being imaged is typically not an option on a standard computer workstation.

Recent advances in medical viewing technology have tried to address some of these limitations by enabling the display of medical images using techniques such as augmented reality, where the object to be visualized is displayed mixed in with real-world imagery [20]. This includes equipment designed for applications such as image-guided surgery [21–23] or other interventions [24], sometimes using equipment such as projectors [25] or cameras [26]. However, such solutions have typically not been designed for data reslicing but rather for displaying of whole anatomies or volumes, and often involve expensive equipment.

Here, we describe ARmedViewer, a medical image viewing app designed for mobile iOS devices using augmented reality technology. The motivation for the app is threefold: 1) Allowing more mobile and easy viewing of medical image data through the use of a mobile device instead of a computer workstation; 2) Enabling very simple and intuitive choice of viewing plane orientation and position (i.e., reslicing) by physical movement of the mobile device in three-dimensional space instead of on a two-dimensional screen; 3) Enabling the user to view the medical images in the context of the imaged anatomy by mixing the image data with real-world images using augmented reality (AR). The benefits of the app were measured through two user experience surveys and through timed user tests.

2. Methods

2.1. Design and functionality of the software program

The program was developed in the Xcode development environment [27] on a MacBook Proin the Swift programming language [28]. The development made use of the SceneKit and ARKit frameworks. The program was compiled and tested on an iPhone XS but should work on any iOS device capable of running SceneKit and ARKit, including iPad.

SceneKit [29] is a 3D graphics framework from Apple for designing 3D scenes in apps. In this framework, a variety of 3D shapes can be drawn with a range of materials and brightness levels. The framework offers a wide variety of other resources not used in this work, including a physics engine and lighting effects.

ARKit [30] is a framework from Apple for building augmented reality applications, mixing 3D graphics with the camera feed of an iOS device. This is achieved with a combination of tracking the location of the mobile device and capturing and processing the surrounding scene, enabling the display of 3D objects on the screen as if they were physically located in the environment.

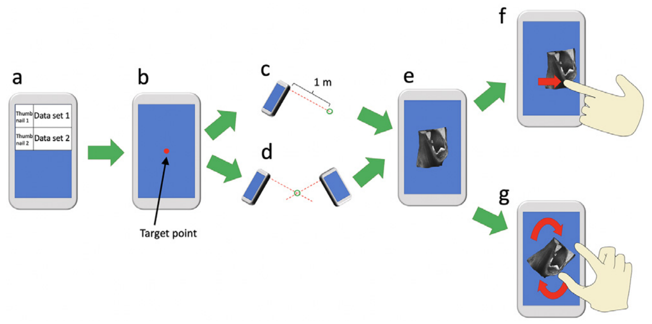

Fig. 1. A schematic of the ARmedViewer workflow. (a) After opening the app, the user is presented with a list of available local data sets. (b) After choosing a data set, the user is shown the camera feed of the mobile device, with a red targeting dot in the center. (c) If the user chooses to immediately load the data, the 3D data set is centered on the targeting dot 1 m away from the device. (d) The user also has the option to fix the targeting dot in 3D space, thereby controlling the depth at which the data set appears. This is done by tapping the phone while orienting it at two different angles. The targeting dot is fixed at the point where the normal vectors to the device during tapping are closest to intersecting. (e) The data is then loaded and appears as a part of the environment using augmented reality technology. (f) The user can shift the data by dragging it with a finger. (g) The user can rotate the data using two fingers.

After opening the app, the user is presented with a list of datasets present on the phone (Fig. 1a). The list includes a name of the dataset and an optional thumbnail image, showing a sample slice. The user chooses a dataset to display by clicking on it in this list. Once a dataset has been selected, the user can press a camera button in the corner of the screen to be taken to the camera view.

In the camera view, a red dot appears at the center of the screen (Fig. 1b). This red dot moves with the screen and is for helping the user accurately place the 3D data to be viewed within its environment. The user has the option of loading the data immediately by pressing a “load data” button. In this case, the data set will appear on the screen, centered on the red targeting dot, 1 m away from the device (Fig. 1c). Alternatively, the user can fix the 3D coordinates of the targeting dot by tapping the screen once, then moving the device and tapping the screen again (Fig. 1d). At each tap, the program calculates the formula for a line projected orthogonally away from the phone to the red dot. The point in 3D space where the two lines come closest to intersecting will then be fixed as the targeting point. The red dot will stay at this point and not move with the device any longer. More tapping of the screen will repeat this procedure, recalculating the targeting point based on the last two taps. Once the user is satisfied with the target point, the data can be loaded by pressing the “load data” button.

Upon loading, the data appears on the screen as a 3D object comprised of multiple 3D pixels, or “voxels”. Each voxel is rendered as a separate 3D rectangular cuboid, with location, dimensions, and signal intensity as described in the data file. This was done using the classes provided by SceneKit, by first creating an SCNBox geometry for each voxel, with the voxel size specified in the input file header. Its RGB values were then set to the intensity value registered in the input file using the UIColor feature, and then creating an SCNNode having this geometry. The SCNNode was then given a position based on its order in the input file, with the red targeting dot serving as the origin point of the coordinate system. No shading or transparency was applied to the individual cuboids. Only the voxels with intensity above 1% of the maximum voxel intensity of the whole data set are rendered. This way, voxels that are outside the anatomy and only contain noise are not rendered (otherwise, the data set would simply appear as a dark box). If the user is not satisfied with the location or orientation of the dataset, it can be

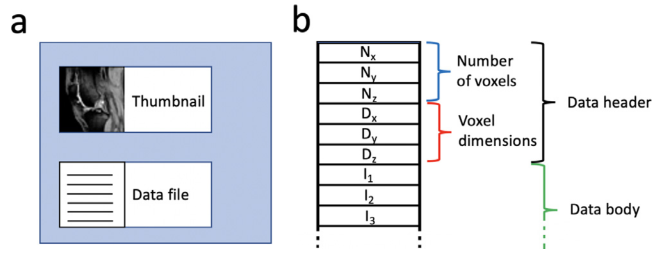

Fig. 2. The data format used for the software. (a) A directory contains a data file and an optional thumbnail. (b) The data file consists of a header, showing the size (N_x, N_y, N_z) of the 3D data matrix (blue) and the voxel dimensions (D_x, D_y, D_z) in μm (red), and a body with several lines corresponding to image voxel intensities (green).

moved by dragging with a finger or rotated by performing a clockwise or counterclockwise two-finger rotation gesture on the screen (Fig. 1f-g). The dragging and rotation take place in the same plane as the mobile device. To drag or rotate the object in a different plane, the device can be moved and reoriented.

A key feature of the program is that it only renders data beyond a fixed distance from the plane of the mobile device. This rendering distance is by default set to 0.5 m, but this can be changed using a slider bar at the bottom of the screen. This has the effect of "slicing" through the data at the given distance, displaying a slice in the same plane as the mobile device. The user can "enter" the anatomy by simply moving the phone closer (or by using the slider bar). In this sense, the mobile device can be thought of as providing a "window" into the anatomy. The resource use of the program, such as battery usage, will depend on the number of voxels rendered and was measured during the testing of the app.

2.2. Data format

The program uses a simple data format as shown in Fig. 2. The baseline data structure is a directory containing a data file and an optional sample thumbnail. When selecting the data from a list of data sets in the app, the thumbnail, if available, is displayed, as well as the name of this data directory. The data file consists of a header and a body. The header contains information about the number of voxels in each dimension (N_x, N_y, N_z) and the dimensions of each voxel. The body consists of a sequence of numbers, corresponding to the signal intensity of the voxel on a scale of 0 to $2^{16}-1$ (going from dark to bright). The data is entered into the file by looping through the z, x, and y coordinates (with z in the outermost loop), and the program automatically reads the data the same way. All the entries in the data file had the format UInt16. During the construction of the data file, the intensities are scaled so that the voxel with the highest signal intensity in the data set is set to this maximum value. This data file is then placed in the directory corresponding to the app in the "Files" directory of the mobile device.

The simplicity of the input data format allows great flexibility in the design of the data pipeline from the medical imaging device to the mobile device. In our implementation, DICOM image files were read by a MATLAB script (on a MacBook Pro) that combined the 2D images into a 3D volume and performed intensity scaling and resizing of the data, as well as creating the thumbnail. The results were then transmitted to the mobile device with an AirDrop connection, where they were placed in the "Files" directory. However, many other implementations of the data preparation are possible, such as creating a Python script installed locally on an MRI scanner that prepares the data and shares it with the device through a cloud-based service.

2.3. User surveys

15 volunteers were asked to undergo two user surveys, comparing the ease of use of ARmedViewer to the commonly used image viewer Horos [4] with its 3D Volume Rendering functionality, which represented the most comparable functionality to the app in commonly available software known to the authors. The inclusion criteria was that the respondents not be users of medical image viewing software such as OsiriX and Horos, as familiarity with one platform over the other would distort the comparison. No medical background was deemed necessary, as the users would not be asked to do any medical diagnosis or to identify anatomy. Two respondents had medical training and two had cell biology training, but did not use medical images in their work.

First, in a custom-made survey using a Likert-type scale, the volunteers were asked to judge the user experience of the following operations on ARmedViewer and Horos: 1) Preparing and loading the data into the software; 2) Opening the data set from within the software; 3) Moving the 3D data set in space; 4) Displaying the data at any given angle (in ARmedViewer, this could be done by rotating the data set or by moving the phone to change the point of view); 5) Displaying a slice through the data set at any location and viewing angle. The volunteers were asked to grade how intuitive it was to learn these operations and how easy they were to perform, once learned, on a scale of 1 (very intuitive/easy) to 5 (very unintuitive/hard).

Next, the same survey respondents were asked to compare the two software platforms using the commonly used Systems Usability Scale survey [31]. This involved 10 statements about the usability of each software platform, which the users could agree or disagree with on a scale of 1 (strongly disagree) to 5 (strongly agree). We refer to Brooke et al. [31] or to Fig. 6c for a list of the usability statements. The users were given instructions on each platform for 10 min and then used each platform on their own for 5 min.

For both surveys, the average and standard deviation of the responses were calculated for both platforms over the 15 respondents. Statistical differences in mean values were calculated using a two-tailed t-test with α=0.05. Care was taken to make the rendering settings as similar as possible between the two platforms. No transparency, shading, or lighting effects were used in either case, making the rendering very simple on both platforms.

2.4. Timed user task

To compare the effectiveness of the app to standard medical image viewing software, a 3D knee MRI dataset, acquired with the double-echo steady-state (DESS) sequence [32–34], was randomly chosen from the open-access osteoarthritis initiative study (OAI) [35]. The image spatial resolution was downsampled to $2.8 \times 2.8 \times 2.8$ mm^3 on both platforms to not exceed the draw call limitations of the device. 3 bright spherical artificial lesions, 7 mm in diameter, were randomly placed within the tissue, as shown in Fig. 3. Two volunteers were asked to obtain a visualization plane that displayed all 3 lesions. This task was repeated 6 times for each volunteer, with different lesion locations each time. For 3 of the 6 datasets, the volunteers did this using the app, while for the other 3, the Horos medical image viewer software, applying 3D Volume Rendering, was used [4]. The time it took the user to obtain this plane was measured for each data set. For one of the users, the timing involved in preparing the data and loading it into the software was also recorded.

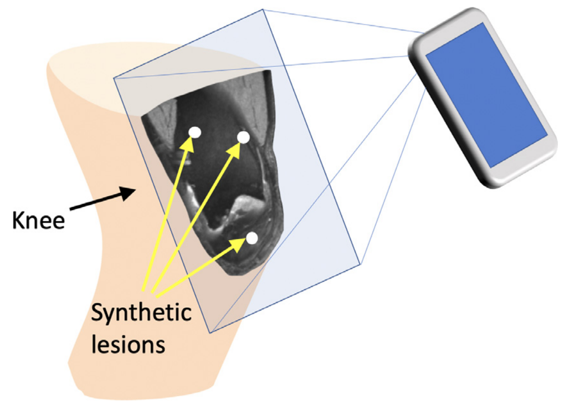

Fig. 3. To assess the ease of using the app, three synthetic lesions were inserted into a knee data set and two users were asked to manipulate the view point to obtain the slice showing all lesions. For each user, this process was done repeated 3 times in the app and 3 times on a conventional medical image viewer on a computer.

3. Results

3.1. Program functionality

Sample images from the program are shown in Fig. 4 (a-i). At first, the user can choose between two data sets located on the mobile device (Fig. 4a). After selecting a data set and pressing the camera button in the top right corner, the camera feed appears (Fig. 4b). A red targeting dot is visible to help position the data. The red dot will remain at the center of the camera view unless it is fixed in space as shown in Fig. 1d, but this is optional. After pressing “Load Data”, the dataset appears in the view (Fig. 4c). If the red dot was fixed in space, the data set will be centered on that 3D coordinate. Otherwise, the data set will appear 1 m away from the device, centered on the red targeting dot. Using the AR capabilities of the mobile device, including motion tracking and scene capturing, the data set appears as a part of the environment and can be viewed from different angles by moving and rotating the device (Fig. 4d). By bringing the device closer to the 3D data, or by using the slider bar at the bottom of the screen, the viewing plane “enters” the data set, parallel to the phone (Fig. 4e). Fig. 4f shows the mobile device from afar, overlaying the data set on the human anatomy. If the data set is placed wrong in the beginning, this can be fixed by dragging and rotating the data set to have it appear at the desired location (Fig. 4g-i).

The battery use of the program, with no other apps in use during the measurements, is shown in Fig. 5. The battery levels of the mobile device dropped substantially while using the program to render a full knee dataset, by 3% within 10 min and by 10% within 20 min. When the program was not in use, the battery levels stayed at 100%. The battery’s maximum capacity was 89% and

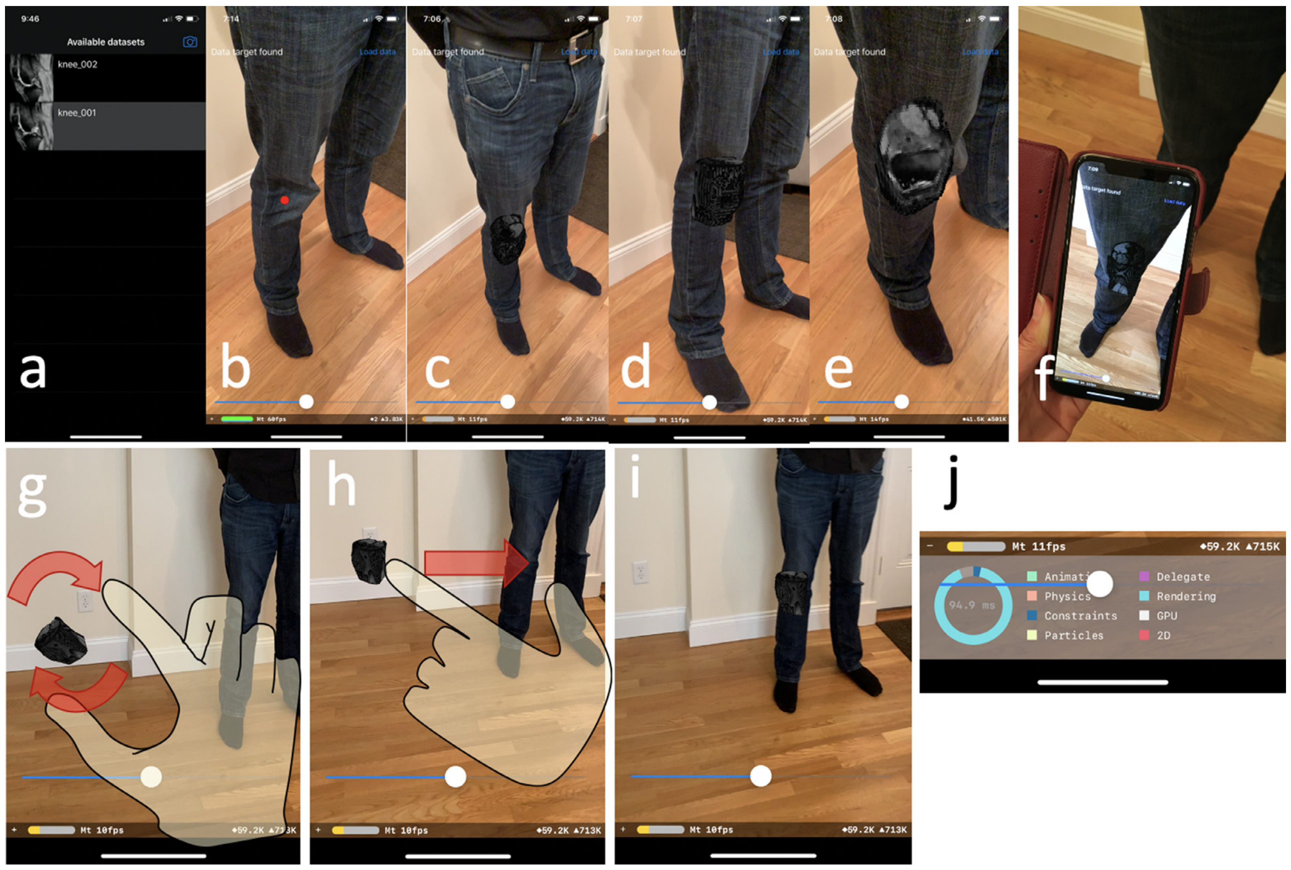

Fig. 4. Sample images from running the app. (a) The user is presented with two knee data sets. (b) The user uses the red targeting dot to set the desired location of the data. (c) Upon loading, the data appears overlaid on the subject. (d) Moving the device around shows the data from different angles. (e) Bringing the device closer allows the user to “enter” the anatomy. The sliding bar at the bottom allows the user to change the distance where this happens. (f) The scene from the point of view of the user. (g) In another instance, the user has placed the data in the wrong spot with the wrong orientation. The user reorients the data using a two-finger rotation gesture. (h-i) Having reoriented the data, the user drags the data to the desired location. Note: Hands and arrows are drawn in panels g-h and are not a part of the screenshot. (j) The information panel at the bottom can be expanded. It shows 11 frames per second, 59,200 draw calls, 715,000 polygons, and a 94.9 ms updating time. Most of this time is dedicated to rendering the voxels.

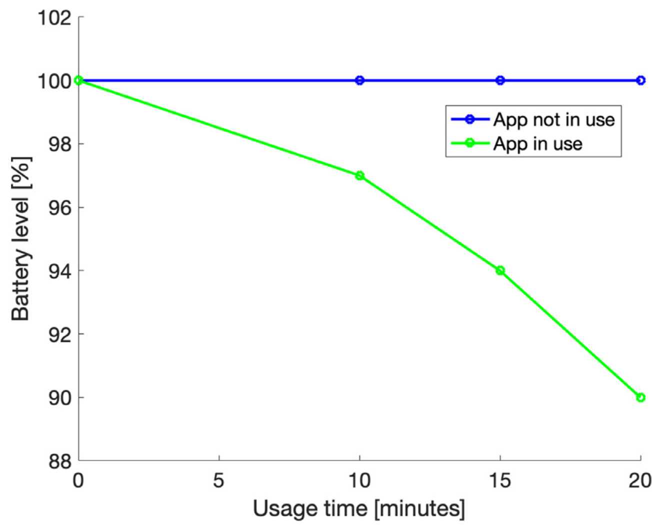

Fig. 5. Battery usage of mobile device while using the program (green), compared to battery levels when app was not in use (blue). No other apps were in use on the device during the time of measurement.

was diagnosed by the device as working at peak performance capacity.

As can be seen towards the bottom of Figs. 4b-d, rendering the full knee volume changed the frames per second (FPS) from 60 to about 10–11, making the video feed less smooth. Draw calls per frame increased from 2 to about 59,200, and the number of polygons drawn increased from about 3800 to about 714,000. As can be seen in Fig. 4e, these numbers scaled with the number of polygons rendered, so when a smaller portion of the volume was rendered by cutting a slice through it, the frame rate increased and the draw calls and polygons decreased.

3.2. User surveys

The user survey responses are displayed in Fig. 6. The custom survey responses are shown in Figs. 6a-b. The respondents reported changing the point of view and freely reslicing the data to be significantly more intuitive to learn using the app, and the reslicing was also significantly easier to do once learned. Other responses were not significant. The mean response for how intuitive and easy it was to load the data was higher (more negative) for the app, but this was not statistically significant.

For the System Usability Scale, ARmedViewer scored significantly better than Horos in the categories of lower system complexity, not needing technical support, most people being able to learn the system quickly, being less cumbersome to use, and not needing to learn many things before getting going with the system. Horos did not score significantly better than ARmedViewer in any category.

3.3. Timed user task

The results of the experiment asking two volunteers to obtain a viewing plane that contained 3 randomly placed lesions is shown in Fig. 7. For user 1, using the workstation the operation took on average 1083 s while using ARmedViewer on a mobile device it took 135 s, giving approximately 8-fold reduction in time. The standard deviation of the measured time using the mobile app was 63 s, while for the computer workstation it was 344 s. For user 2, average times were 622 s for the workstation and 65 s for ARmedViewer, a 10-fold reduction in time, with standard deviations of 307 and 20 s, respectively. For user 2, the average time for preparing and loading the data was on average 42 s for the app and 20 s for the laptop, with standard deviations of 5 s and 4 s, respectively. This was not measured for user 1.

4. Discussion

We have presented ARmedViewer, an augmented reality app that allows for visualization of medical imaging data on an iOS mobile device. This presents an inexpensive and portable option for medical image data analysis. The program offers the ability to choose the desired scan plane with no prior technical expertise. We have demonstrated that selecting a randomly determined viewing plane is considerably faster using the program than with conventional medical image viewing software, and some functionalities were measured as significantly more intuitive and easy to use than on a standard medical image viewer. Furthermore, the program allows the end user to view the medical image data overlaid with the real-world anatomy. This functionality could enable a more accurate determination of the location of any pathologies relative to the exterior of the body.

In the custom user survey, the volunteers reported that learning to change the field of view and reslicing the data (freely selecting an image plane cutting through the data at any position and angle) was significantly ($p < 0.01$) more intuitive to learn in ARmedViewer than in Horos. Reslicing the data was also reported as significantly ($p < 0.05$) easier to do once learned in ARmedViewer. No significant difference was reported for moving the data in space, which is not surprising, since moving the data is expected to be a simple operation. This confirms, however, that moving the data with ARmedViewer to align the medical data with the real-world anatomy is straightforward. No significant difference was either reported in terms of the ease of loading and opening data on the two platforms. On average, respondents described more difficulty in loading data onto the mobile platform, which is not unexpected since this involved using Apple's Airdrop functionality as an extra step, but this difference was not significant.

A potential explanation for the users reporting changing viewing angles and reslicing to be significantly more intuitive and the reslicing to be significantly easier on the mobile device, as well as the reslicing being faster, is that these are effectively 3D operations, which can be challenging to perform on the 2D screen of a computer workstation. Using the mobile device, these operations are performed using hand gestures in true 3D space, which might feel more natural to the user. The app therefore provides functionality that workstation software cannot provide, in spite of the higher computational power of the workstation. The 3D rendering functionality of Horos (and OsiriX) was the closest functionality to this that the authors were aware of, but the two user experiences are unavoidably different for these reasons. Providing this new user experience was a large part of the motivation for this study.

In the custom survey, a Likert-type scale specifically designed for this program was used. This type of scale was used as it enabled user ratings on an easily understandable scale. Using a specially designed scale allowed questions better tailored to the functionality of the program. The more standardized System Usability Scale (SUS) [31] part of the survey allowed for responses that could be more easily compared to user experiences of other programs. The two approaches therefore complement one another and provide ratings that both examine the specific features of the program as well as ratings that can be evaluated in a larger context with different software.

The measurements of the time required to obtain a randomly oriented slice through the 3D data set demonstrated an approximately 9-fold average time reduction using the mobile app compared to conventional medical image viewing software on a computer workstation. Additionally, the standard deviation in the time duration decreased by an average factor of 8. This indicates that

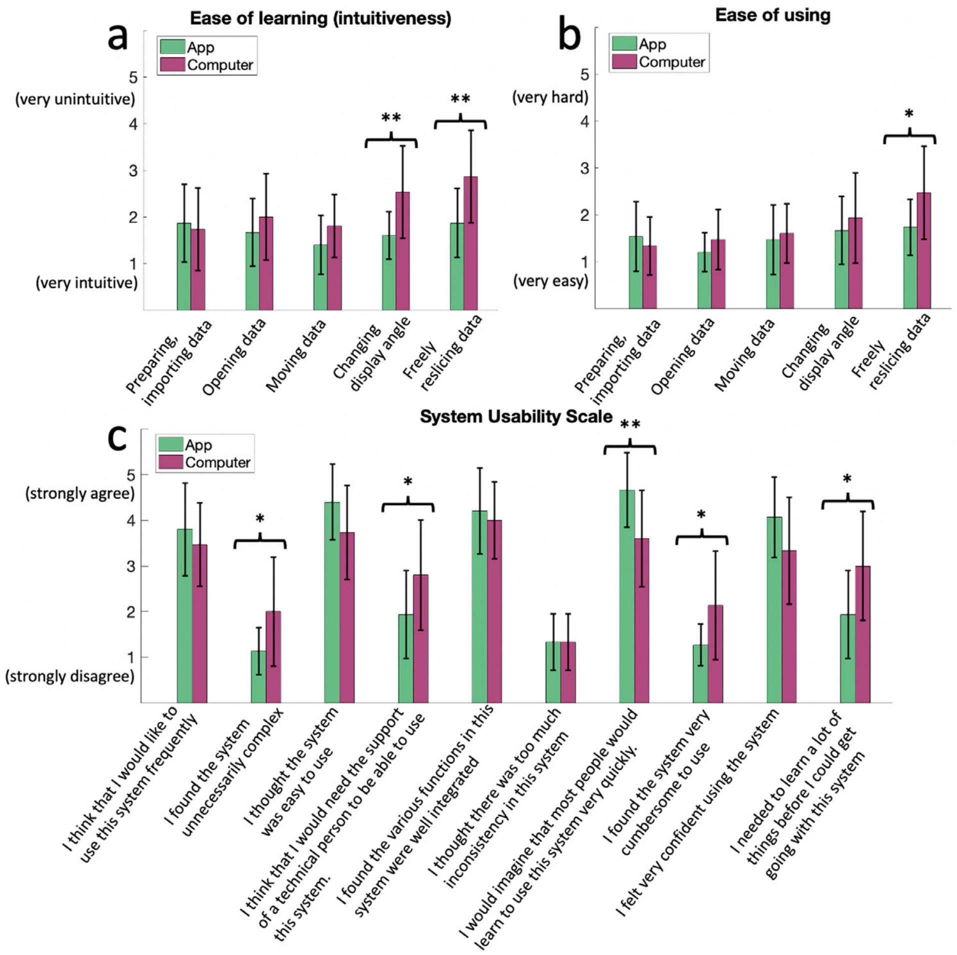

Fig. 6. Results from app-specific survey and system usability scale survey of 15 respondents. Bar values show mean result, error bars show standard deviation for the app (green) and the computer workstation (purple). Significant differences between means are labeled with * ($p < 0.05$) and ** ($p < 0.01$).

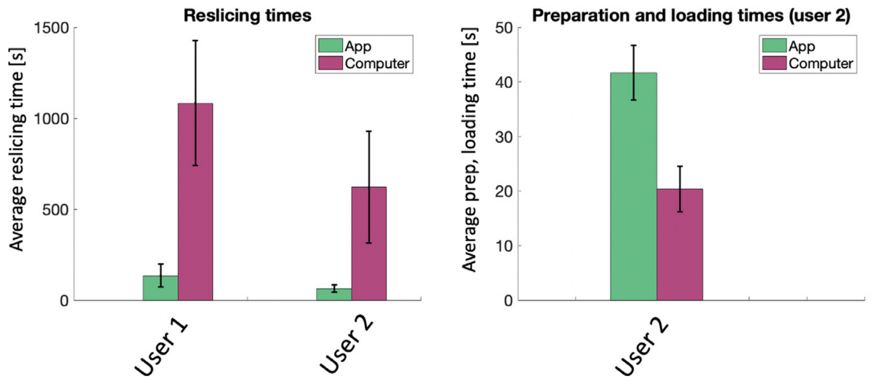

Fig. 7. (a) Results from measuring the time it took for two users to obtain a slice showing three synthetic lesions as shown in Fig. 3 using the app (green) and using a computer workstation (purple). The app clearly works faster and has less variability in how long it takes to obtain the correct view. (b) Comparisons of the time it took to prepare and load the data into each platform for one of the users. The measured time was longer on average for the app.

not only is choosing a viewing plane on average faster using the app, but the time taken is also more consistent. This could be of value in planning workflow involving several data sets.

Other studies have demonstrated the usefulness of viewing medical image data using augmented reality devices such as the Hololens [23,36,37]. However, the design of ARmedViewer, being targeted at mobile devices, presents unique opportunities. A key feature of the program is to determine the depth and angle of the viewing plane by only rendering voxels at a certain distance from the plane of the device. This allows reslicing of the data by simple physical movements of the device, resulting in a very intuitive user experience. Furthermore, using a mobile device has the added benefit of greatly reducing hardware costs associated with the program, as ownership of smart mobile devices is currently very widespread. This could allow use of the software in poorer regions or developing countries.

ARmedViewer can be useful for situations where the operator needs to visualize medical imaging data quickly in a mobile manner. This could involve battlefield scenarios, where medical personnel need to perform triage, or emergency room situations. In recent years, much progress has been made in building portable medical imaging devices that are well suited for such purposes, such as low-field portable MRI scanners [11,13–16]. The ability to visualize the imaged anatomy in a fast and intuitive way could complement such equipment well. This visualization methodology is well suited for such low-field, portable MRI scanners in additional ways, as they often have a more open design. In a standard clinical MRI scanner, the anatomy being scanned is not easily visible except along the scanner bore, and the strong magnetic field could potentially affect the mobile device. In such a system, the patient would therefore likely need to be rolled out of and away from the scanner before using the software. In an open-design, low-field system, the patient can be examined with the app while lying in the scanner. It should also be noted that the program can be used with other medical imaging technologies than MRI, such as CT.

The presented work has certain limitations. As each voxel in the image data set is rendered as a separate cuboid, large data sets put strains on the rendering capabilities of the device and eventually lead the program to freeze. The current version of the software tested on an iPhone XS was able to load data from approximately 130,000 voxels, which is slightly more than 50×50×50 resolution, with approximately 60,000 of them rendered after noise thresholding. Debugging showed the software to stall during rendering and not loading, indicating that the limiting factor in the number of voxels appeared to be the number of draw calls processed by the renderer, and not the size of the input file. As the graphical processing capabilities of mobile devices continue to grow with each device generation, this limit can be expected to increase. As demonstrated, the program also can also substantially drain the device battery, even with peak battery performance capacity. Devices with less battery capacity could therefore potentially not use the program for long periods of time. The only hard limit for the iOS device to run the software is to be AR-compatible with the first generation of ARKit, which includes any iPhone after and including the SE and 6 s series, and iPads (or iPad minis) and iPad Airs of 5th and 3rd generation or later, respectively, or any generation of iPad Pro. Also, the anatomical datasets used for the user tests were all knee datasets. Future work could involve comparing user experiences for different anatomies. Furthermore, the program was designed only for Apple devices, using ARKit and Scenekit. This was done as the purpose of this study was to show the feasibility and benefits of using hand-held augmented reality for viewing and reslicing of medical images, which only needed to be done on one platform as a proof-of-concept. However, as part of future work to widen the user base of the software, alternate versions could potentially be built for Android devices using environments such as Google's ARCore, ViroCore, or Unity.

5. Conclusion

ARmedViewer, a custom-made app for viewing medical data on an iOS device, allows for portable, immersive viewing of medical image data with unparalleled ease of choosing the data viewing plane.

Declaration of Competing Interest

M.S.R. is a co-founder of Hyperfine Research, Inc. and BlinkAi, Inc.

Acknowledgements

This work was supported with DARPA 2016D006054.

References

[1] P. Mildenberger, M. Eichelberg, E Martin, Introduction to the DICOM standard, Eur Radiol 12 (4) (2002) 920–927, doi:10.1007/s003300101100.
[2] M. Larobina, L Murino, Medical image file formats, J Digit Imaging 27 (2) (2014) 200–206, doi:10.1007/s10278-013-9657-9.
[3] A. Rosset, L. Spadola, O Ratib, OsiriX: an open-source software for navigating in multidimensional DICOM images, J Digit Imaging 17 (3) (2004) 205–216, doi:10.1007/s10278-004-1014-6.
[4] https://horosproject.org/. Retrieved June 2020.
[5] https://www.radiantviewer.com/. Retrieved June 2020.
[6] E. Siegel, B. Reiner, M. Abiri, et al., Filmless radiology reading room: a survey of established picture archiving and communication system sites, J Digit Imaging 13 (2) (2000) 22–23, doi:10.1007/bf03167618.
[7] S. Waite, S. Kolla, J. Jeudy, et al., Tired in the Reading Room: the Influence of Fatigue in Radiology, J Am Coll Radiol 14 (2) (2017) 191–197, doi:10.1016/j.jacr.2016.10.009.
[8] K. Peace, E.M. Wilensky, S. Frangos, et al., The Use of a Portable Head CT Scanner in the Intensive Care Unit, J Neurosci Nurs 42 (2) (2010) 109–116.
[9] D.M. Becker, C.A. Tafoya, S.L. Becker, G.H. Kruger, M.J. Tafoya, T.K Becker, The use of portable ultrasound devices in low- and middle-income countries: a systematic review of the literature, Trop Med Int Heal 21 (3) (2016) 294–311, doi:10.1111/tmi.12657.
[10] M. Sarracanie, C.D. Lapierre, N. Salameh, D.E.J. Waddington, T. Witzel, M.S Rosen, Low-Cost High-Performance MRI, Sci Rep 5 (2015) 1–9, doi:10.1038/srep15177.
[11] https://hyperfine.io. Retrieved June 2020.
[12] C. Zimmerman Cooley, M.W. Haskell, S.F. Cauley, et al., Design of sparse halbach magnet arrays for portable MRI using a genetic algorithm, IEEE Trans Magn 54 (1) (2017), doi:10.1109/TMAG.2017.2751001.
[13] C.Z. Cooley, J.P. Stockmann, B.D. Armstrong, et al., Two-dimensional imaging in a lightweight portable MRI scanner without gradient coils, Magn Reson Med 73 (2) (2015) 872–883, doi:10.1002/mrm.25147.
[14] P.C. McDaniel, C.Z. Cooley, J.P. Stockmann, L.L Wald, The MR Cap: a single-sided MRI system designed for potential point-of-care limited field-of-view brain imaging, Magn Reson Med 82 (5) (2019) 1946–1960, doi:10.1002/mrm.27861.
[15] L.L. Wald, P.C. McDaniel, T. Witzel, J.P. Stockmann, C.Z Cooley, Low-cost and portable MRI, J Magn Reson Imaging (2019), doi:10.1002/jmri.26942.
[16] https://promaxo.com. Retrieved June 2020.
[17] E.W. Akins, J.A. Hill, J.R. Fitzsimmons, C.J. Pepine, C.M Williams, Importance of imaging plane for magnetic resonance imaging of the normal left ventricle, Am J Cardiol 56 (4) (1985) 366–372, doi:10.1016/0002-9149(85)90866-5.
[18] J.J. Hermans, A.Z. Ginai, N. Wentink, W.C.J. Hop, A Beumer, The additional value of an oblique image plane for MRI of the anterior and posterior distal tibiofibular syndesmosis, Skeletal Radiol 40 (1) (2011) 75–83, doi:10.1007/s00256-010-0938-9.
[19] S.B. Gay, N.C. Chen, J.J. Burch, T.R. Gleason, A.M Sagman, Multiplanar Reconstruction in Magnetic Resonance Evaluation of the Knee, Invest Radiol 28 (2) (1993) 142–145, doi:10.1097/00004424-199302000-00011.
[20] M. Mikhail, K. Mithani, G.M Ibrahim, Presurgical and Intraoperative Augmented Reality in Neuro-Oncologic Surgery: clinical Experiences and Limitations, World Neurosurg 128 (July 2003) (2019) 268–276, doi:10.1016/j.wneu.2019.04.256.
[21] N. Bourdel, T. Collins, D. Pizarro, et al., Augmented reality in gynecologic surgery: evaluation of potential benefits for myomectomy in an experimental uterine model, Surg Endosc 31 (1) (2017) 456–461, doi:10.1007/s00464-016-4932-8.
[22] Chen J gang, Han K wei, Zhang D feng, Li Z xing, Li Y ming, Hou L jun, Presurgical Planning for Supratentorial Lesions with Free Slicer Software and Sina App, World Neurosurg 106 (2017) 193–197, doi:10.1016/j.wneu.2017.06.146.
[23] S.L. Perkins, M.A. Lin, S. Srinivasan, A.J. Wheeler, B.A. Hargreaves, B.L Daniel, A Mixed-Reality System for Breast Surgical Planning, Adjun Proc 2017 IEEE Int Symp Mix Augment Reality, ISMAR-Adjunct 2017 (2017) 269–274, doi:10.1109/ISMAR-Adjunct.2017.92.
[24] A. Mewes, F. Heinrich, B. Hensen, F. Wacker, K. Lawonn, C Hansen, Concepts for augmented reality visualisation to support needle guidance inside the MRI, Healthc Technol Lett 5 (5) (2018) 172–176, doi:10.1049/htl.2018.5076.
[25] A. Mewes, F. Heinrich, U. Kägebein, B. Hensen, F. Wacker, C Hansen, Projector-based augmented reality system for interventional visualization inside MRI scanners, Int J Med Robot Comput Assist Surg 15 (1) (2019) 1–9, doi:10.1002/rcs.1950.
[26] D. Inoue, B. Cho, M. Mori, et al., Preliminary study on the clinical application of augmented reality neuronavigation, J Neurol Surgery, Part A Cent Eur Neurosurg 74 (2) (2013) 71–76, doi:10.1055/s-0032-1333415.
[27] https://developer.apple.com/xcode/. Retrieved June 2020.
[28] https://developer.apple.com/swift/. Retrieved June 2020.
[29] https://developer.apple.com/scenekit/. Retrieved June 2020.
[30] https://developer.apple.com/documentation/arkit. Retrieved June 2020.
[31] J. Brooke, SUS: a "Quick and Dirty" Usability Scale, in: PW Jordan, B Thomas, IL McClelland, B Weerdmeester (Eds.), Usability Evaluation in Industry, Taylor & Francis, London, 1996, pp. 189–194.
[32] H. Bruder, H. Fischer, R. Graumann, M Deimling, A new steady-state imaging sequence for simultaneous acquisition of two MR images with clearly different contrasts, Magn Reson Med 7 (1) (1988) 35–42, doi:10.1002/mrm.1910070105.
[33] T.W. Redpath, R.A Jones, FADE–A new fast imaging sequence, Magn Reson Med 6 (2) (1988) 224–234, doi:10.1002/mrm.1910060211.

[34] S.Y. Lee, Z.H. Cho, Fast SSFP gradient echo sequence for simultaneous acquisitions of FID and echo signals, Magn Reson Med 8 (2) (1988) 142–150, doi:10.1002/mrm.1910080204.

[35] F. Eckstein, W. Wirth, M.C Nevitt, Recent advances in osteoarthritis imaging - The Osteoarthritis Initiative, Nat Rev Rheumatol 8 (10) (2012) 622–630, doi:10.1038/nrrheum.2012.113.

[36] S. Sirilak, P. Muneesawang, A New Procedure for Advancing Telemedicine Using the HoloLens, IEEE Access 6 (2018) 60224–60233, doi:10.1109/ACCESS.2018.2875558.

[37] R. Vassallo, A. Rankin, E.C.S. Chen, T.M Peters, Hologram stability evaluation for Microsoft HoloLens, Med Imaging 2017 Image Perception, Obs Performance, Technol Assess 10136 (March 2017) (2017) 1013614, doi:10.1117/12.2255831.