AWARD NUMBER: W81XWH-19-C-0131

TITLE: Medic's Enhanced Diagnostic Imaging Companion (MEDIC) System Phase I Final Report

PRINCIPAL INVESTIGATOR: Alexander Pavloff

CONTRACTING ORGANIZATION: Augmntr, Inc.

REPORT DATE: February 2020

TYPE OF REPORT: SBIR Phase 1 Final Report

PREPARED FOR: U.S. Army Medical Research and Development Command

Fort Detrick, Maryland 21702-5012

DISTRIBUTION STATEMENT: Approved for Public Release;

Distribution Unlimited

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
Feb 2020	SBIR Phase I Final Report	8 July 2019 – 7 Feb 2020

4. TITLE AND SUBTITLE

The Medic's Enhanced Diagnostic Imaging Companion (MEDIC) System

Phase I Final Report

5a. CONTRACT NUMBER

5b. GRANT NUMBER
W81XWH-19-C-0131

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Alexander Pavloff

Andrew Konopacki

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Augmntr
149 Crown Point Ct, Suite C
Grass Valley, CA 95945

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Medical Research and Development Command
Fort Detrick, Maryland 21702-5012

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited **(SBIR Data Rights, Feb 2020)**

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The purpose of the MEDIC (Medic's Enhanced Diagnostic Imaging Companion) system is to provide combat medics, via Augmented Reality (AR), a digitally-superimposed avatar representing the internal anatomy of their Warfighter patient. Phase I of this SBIR project consisted of a Systems Engineering activity that drove requirements, followed by an implementation of demonstration-level software into a commercial AR headset and supporting hardware. The system was demonstrated inside and outside Augmntr's offices, and the Phase I results indicate the concept is feasible and valuable.

15. SUBJECT TERMS

Augmented Reality, Avatar, Medic, IVAS, HoloLens

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			USAMRMC
Unclassified	Unclassified	Unclassified	Unclassified	26	19b. TELEPHONE NUMBER *(include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

Table of Contents

1 Introduction

Augmntr's Medic's Enhanced Diagnostic Imaging Companion (MEDIC) System is a HoloLens-based software application that will allow for real-time viewing of a USARIEM developed Avatar properly overlaid upon a real subject. Augmntr's Phase I research effort was to prove the feasibility of a complete end-to-end system running on the most appropriate hardware available. Augmntr's Phase I effort was successful, with a complete prototype system being demoed outside of Augmntr's offices.

2 Overall Project Summary

2.1 Project Goals

From the beginning of the project, Augmntr's goal was to get a representative system running in a HoloLens in a prototype manner. These performance concerns for deployment to mobile hardware like the HoloLens now and in the future were in the forefront of development, so all nearly all software developed was written in C#, targeted for the Unity 3D engine or Windows.

The Unity "game" engine[1] is the primary development environment for Augmented Reality (AR) and Virtual Reality (VR) software, with direct support for all major hardware devices available today. While originally designed for games only, the advent of VR and AR computing has resulted in heavy use of it in non-gaming applications. "Standard" 2D software, whether browser-based (websites), desktop-based (Microsoft Office, etc), or scientific-focused (MATLAB, Jupyter, etc) does not take advantage of the spatial element of modern AR devices, where three-dimensional content and controls can be placed in the real world, moved around, and interacted with.

The other major engine, Unreal[2], is also a solid development choice, but only recently added support for the HoloLens, and requires coding in lower-level C++ vs the more productive C# of Unity.

This report groups the work done on the project by task/topic, not chronologically.

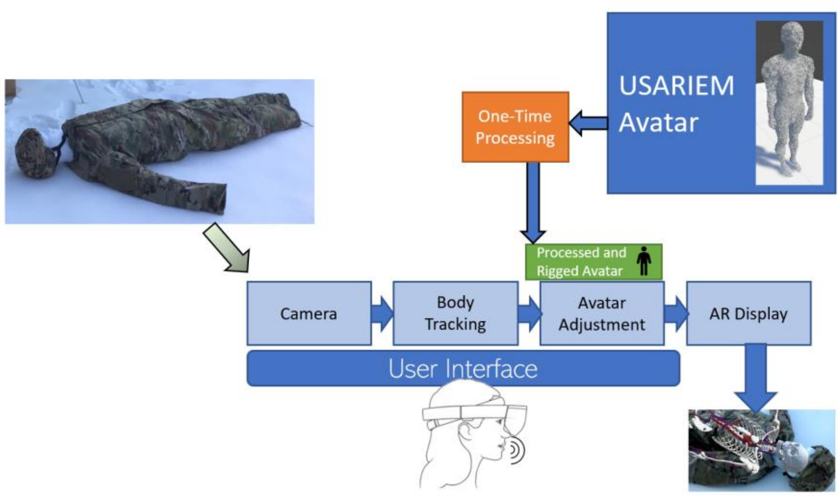


Figure 1: Overall MEDIC concept

[1] https://unity.com/

[2] https://www.unrealengine.com/en-US/

2.2 Avatar

USARIEM provided Avatar data for use in this project. 5mm data was initially provided, with 2mm data a few months into the project. This data was provided in 2 files. The first was a list of vertices, and the second a list of triangles (referencing the vertices) and the organ that each triangle corresponded too.

For the initial work, all steps were performed manually, with notes taken to look ahead to the future automatic import of warfighter-specific avatars.

This data is close to the OBJ format, which is natively supported by Unity, so some utility code was written to rearrange the data into the OBJ format. Each organ was turned into a separate object, all parented to a single top-level object.

This OBJ file was then loaded into the open-source 3D Modeling Toolkit "Blender"[3] for additional work.

The origin of the model was placed between the feet and under the heels, and the correct orientation was set to match future import into Unity, with Z-Forward Y-Up as the orientation.

Once these steps were performed in Unity, the completed OBJ was saved and ready to import into Unity.

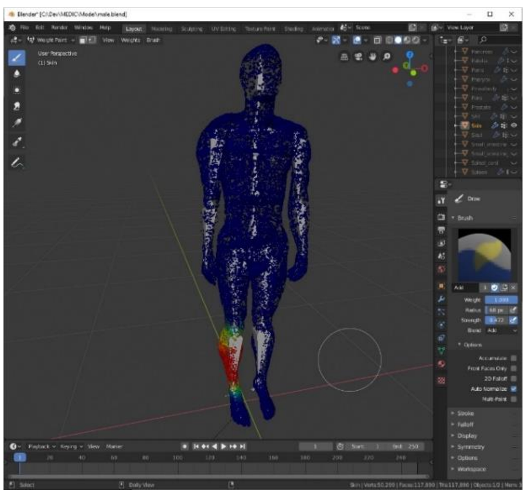


Figure 2: Avatar loaded into Blender

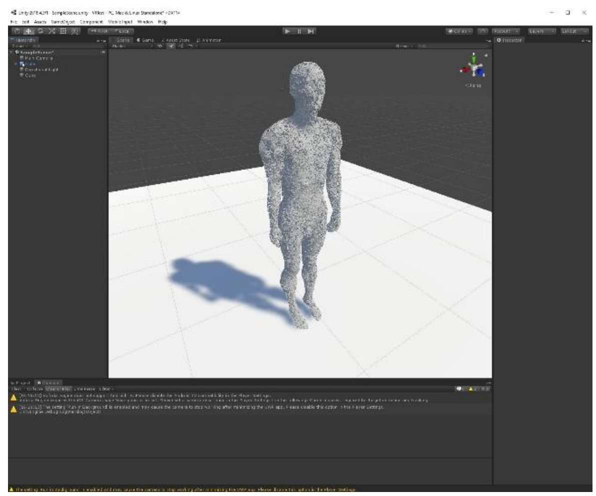


Figure 3: Avatar loaded into Unity

Once imported into Unity, a basic flat mobile-friendly shader was applied. Shaders are used to color in triangles, and a flat shader is just a simple, single color way to color them. The model was placed into the Unity scene, the application was compiled for the HoloLens 1, and basic performance was checked. For AR applications, in order to provide a satisfactory experience for the wearer, framerate must be at least 30 frames-per-second, preferably 60 frames-per-second. This is because unlike a standard 2D application, the "camera", that is, the head of the wearer, is constantly moving. A low-frame rate on an AR application will result in content in the world "jumping" and not staying registered.

For the initial run, displaying every single triangle on every organ resulted in the frame rate falling far below the acceptable 30Hz level. Since the best way to optimize models for display is simply not show

[3] https://www.blender.org/

them, a few combinations of organs were created. Showing these subsets, such as “exterior features” or “skeleton” allowed the application to run at the desired 60Hz.

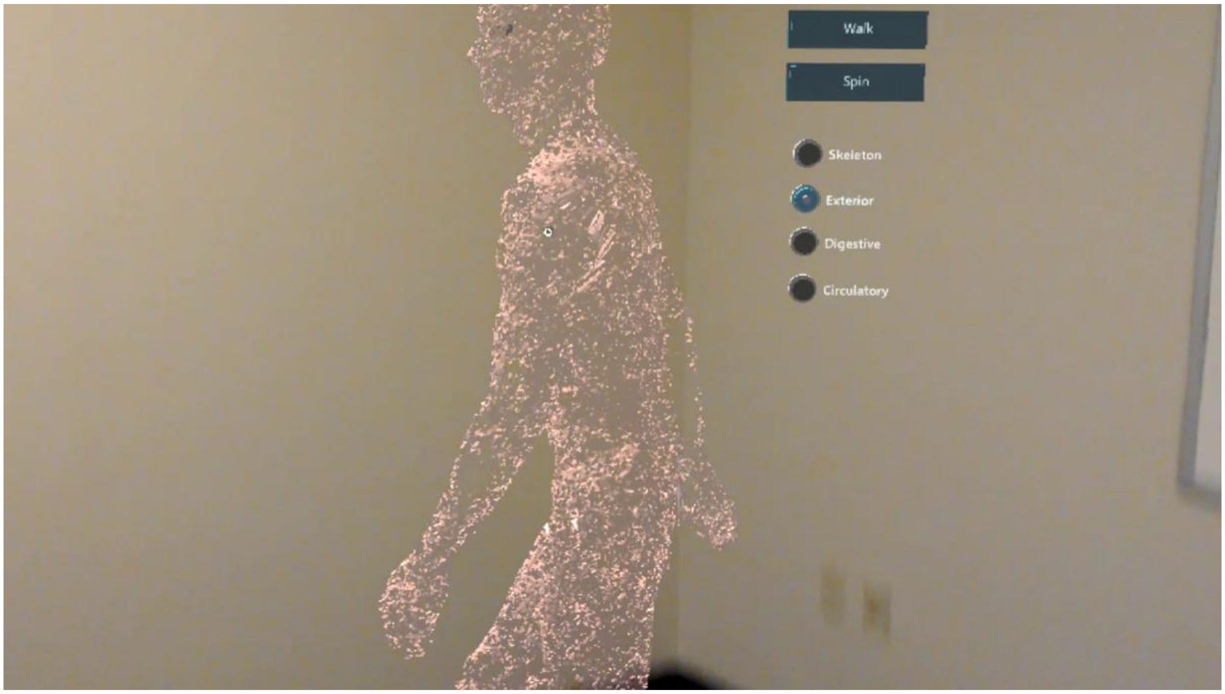


Figure 4: Exterior Subset of 5mm Avatar running in the HoloLens

Augmntr used the 5mm Avatar for the first part of the project, and after receiving the 2mm Avatar, investigated how to improve the visual quality of the Avatar while maintaining performance on the HoloLens. The initial work with the 5mm Avatar used the raw triangles only, and while effective and easy to work with, this avatar was low-resolution and missing features. The 2mm avatar is more complete, with less voids, but has substantially more data, which made it impossible that performance goals would be met using a simple “display all the triangles” method for display and processing on the HoloLens.

	5mm Avatar	2mm Avatar	Comparison
Vertices	160038	2153214	13.45x
Triangles	825734	7144346	8.65x

Figure 5: 5mm vs 2mm comparison

Microsoft gives rough guidelines of ~100K triangles to display for both the HoloLens 1 and 2. The HoloLens 2, while substantially more powerful than the HoloLens 1, has the same pixel density with ~2.3x the area to display due to its increased field of view.

Microsoft’s recommendation is difficult rendering situation such as above be handled by purchasing Microsoft Azure cloud-based computing time and using it to Remote Render to the HoloLens 2. This, however, is not possible for the MEDIC based system due to requirement to avoid cloud-based computing.

So, since local processing and display was required, a variety of techniques was explored to balance performance and overall look, all using features built into Blender and amenable to future automated processing.

Firstly, the 2mm Avatar was imported, and rather than bare triangles, an overall “UV” map was created. A UV Map is a common 3d modelling process used to project a 2D image (also known as a texture) onto a 3D model’s surface.

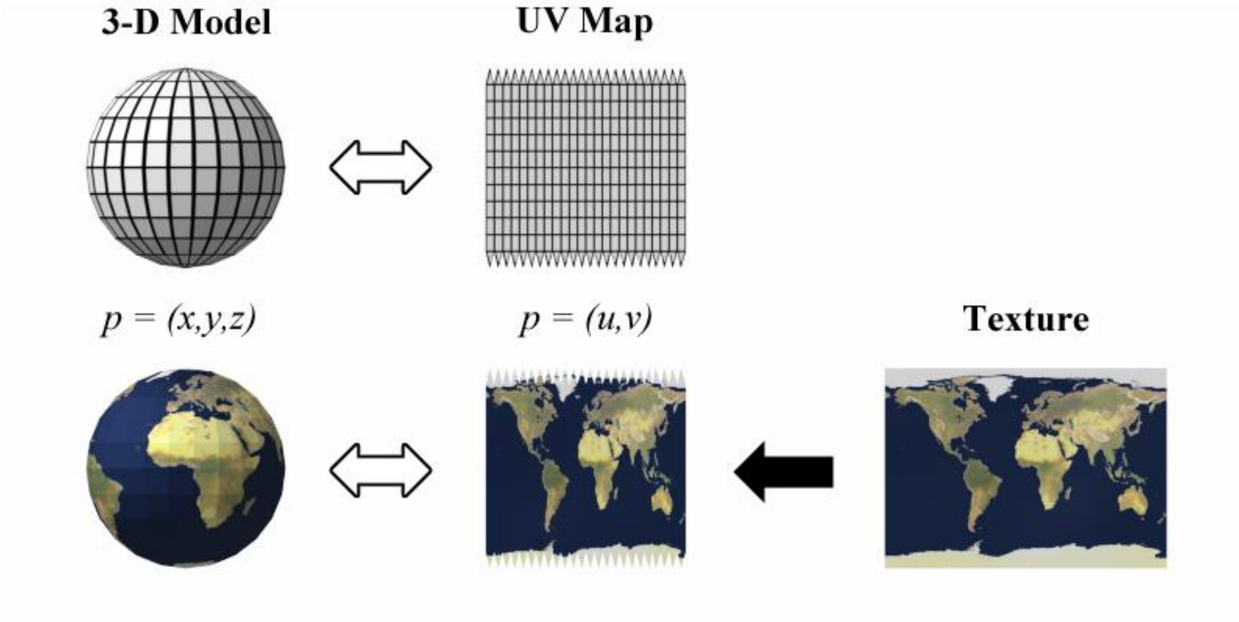


Figure 6: The application of a texture in the UV space related to the effect in 3D[4].

Each “organ” in the Avatar greater than 10-20k triangles was then simplified into the ~10k-40k region. These simplification process combines multiple smaller triangles that are “mostly flat” into a single triangle, greatly reducing the amount of data while maintaining data.

The original UV map was then “smoothed”, the “simplified” avatar was re-rigged, and then the entire system was reloading into Unity and placed into the HoloLens. The result was a high-performance, yet good-looking Avatar.

[4] By Tschmits - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=4568787

Figure 7: From Left to Right: Finished Avatar Skeleton, raw 2mm Avatar, raw 5mm Avatar

In addition to simplification, it is extremely unlikely that all organs in the avatar will be displayed simultaneously, as the amount of data will be very hard to see and use.

Other concepts for increasing detail while reducing data were attempted, such as bump-mapping, were investigated, but not suitable for the display of the organic avatar due to the lack of sharp edges.

All steps of the avatar process were done manually for this phase, but options for automation were reviewed and collected for Phase II.

2.3 Rigging

Rigging is the process of taking a 3D model and adding a "skeleton" to it, such that it can be deformed and moved. This is needed in order to deform the Avatar to match the real-world warfighter. This is generically referred to in the gaming space as "animation", as the usual usage is to have characters "walk" or "climb". The end MEDIC system will not use pre-canned animation, but the same basic pre-existing systems were used for Phase I with Blender and Unity, and so the term "animation" will be used.

Humanoids models are commonly used in games, so prototyping was relatively easy. This rigging utilizes the concept of "Bones", which combined with "animations", allow for interchangeability of models and animations. The Avatar was configured as a standard "humanoid" avatar, and the various triangles corresponding to each "bone" were manually assigned to bones. The resulting rigged Avatar and animations were loaded into Unity. Various animations, such as "walking", "dancing", and "exercising" were tested, allowing iteration on the rigging to ensure all vertices were rigged.

The process of rigging assigns weights to the vertices with respect to the bones and their movement. This is done in two steps. First, Blender runs its auto-rig algorithm to get it mostly set up. Then it is manually processed to make sure the weights are correct. The manual process involves a bone and painting a heatmap of the weight of influence onto the vertices for it. This is iterated over each bone for their own influence. Hard and rigid surfaces such as skull and bones get full weight only to one virtual bone. Softer or stretchy surfaces get the weight spread between two or more bones to show how it moves with the bend to ensure proper deformation in animation.

Rigging is normally applied over an avatar in the "T-pose", which is standing with arms straight out. Because the arms in the delivered avatar were by the sides, the algorithm for automatically determining weights incorrectly assigns parts of the arms to the body and vice versa. By posing the avatar limbs out to the sides, it becomes visually apparent which vertices have incorrect assignment, as it will appear to 'pull' the mistaken parts with them.

Real-world avatars are also not symmetric, unlike most models used in 3D engines. For generated avatars, it is often easiest just to mirror the rigging from one side to the other.

It is important to ensure that coordinates and rotations are correctly assigned and consistent with all tools used. Since animations in are just a list of positions for the bones, so if the coordinates don't match, the animations will behave poorly once imported into Unity.

When importing, you can tell Unity that it is a humanoid rig and assign the bones to locations. Using that, it will try to match up animations to different models. This does not fix coordinate system differences, but only make a specific animation play on an assigned bone. This will also stretch/shrink parts along the long axis to match the input data.

This is a very simplistic approach to animating the movement. The real organs will shift, pull, and move over the others creating a much more complex set of deformation. Weight painting is enough for most movement, but there may be cases where it creates an odd crease. For more accuracy methods such as using corrective shape keys could be used.

All this information will be used to improve the accuracy and automatically import items for Phase II.

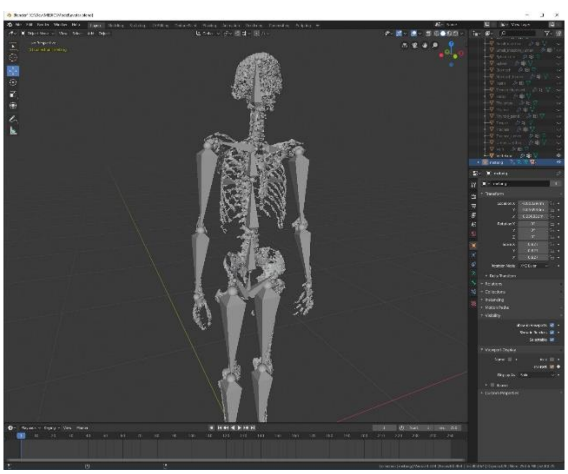


Figure 8: Rigged Avatar

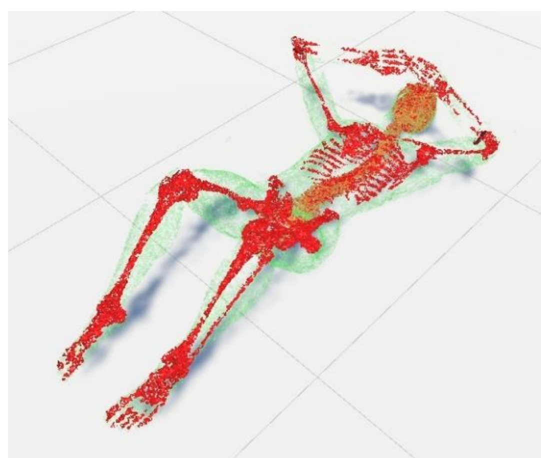


Figure 9: Exercising Avatar Rigging

Future Phases will have to near-automatically perform rigging on Avatars, as spending a few hours per Avatar will not be viable. Ideally this will be performed at an early stage of the Avatar processing before it hits the device, so as to reduce the load time and performance impact on the headset.

2.4 3D Model Formats

After all the Avatar work was done using the tools, it became clear that not all 3D model formats were equal, with substantial difference in capabilities between them. The various formats were compared to find the best, future looking option from capability, standards compliance, and tools support in Unity, Blender, SDKs, and the like.

Importantly, any file format selected should be static throughout the entire Avatar process – from initial creation all the way to final display. Software tools and SDKs that run on a PC for pre-processing and the like, as well as in a mobile device are preferred, as this will enable maximum flexibility without the need for format conversions.

2.4.1 Summary Table

Format	Initial Purpose	License Type	Supports Animation	Unity Support	SDK Availability
OBJ	Simple Modeling	De-facto standard	No	Native	Yes
FBX	Interchange Format	Proprietary License for SDK	Yes	Native	C++ from Autodesk
DAE	Interchange Format	Open, Royalty-Free Standard (last release 2008) by Khronos Group	Yes	Native	Few implementations, mostly abandoned
BLEND	File Format for Blender	Open Source Code	Yes	Native	As part of Blender
glTF	Interoperability and Runtime Efficiency	Open, Royalty Free Standard by Khronos Group	Yes	Open-source importers through SDK	Multiple available for a variety of languages provided by Khronos Group
STL	Stereolithography	De-facto standard	No	No	Various Open Source
3MF	3D Manufacturing	Open, Royalty Free Standard	No	No	Open Source C++ Library
PLY	3D Scanners	Open Standard	No	No	Various Open Source
USD	3D Animation Pipelines	Open Standard	Yes	Preview	Unity implementation is open source

Unity supports additional proprietary formats, such as .3DS (3D Studio) and .DXF (Autodesk), but these are legacy formats being superseded by FBX and glTF. CAD formats such as STEP and IGES were also not evaluated due to being unsuitable for displaying the human form.

2.4.2 OBJ

OBJ (or .OBJ) is a geometry definition file format first developed by Wavefront Technologies for its Advanced Visualizer animation package. The file format is open and has been adopted by other 3D graphics application vendors.

The OBJ file format is a simple data-format that represents 3D geometry alone — namely, the position of each vertex, the UV position of each texture coordinate vertex, vertex normals, and the faces that make each polygon defined as a list of vertices, and texture vertices. Vertices are stored in a counterclockwise order by default, making explicit declaration of face normals unnecessary. OBJ coordinates have no units, but OBJ files can contain scale information in a human readable comment line[5].

2.4.3 FBX

FBX (Filmbox) is a proprietary file format developed by Kaydara and owned by Autodesk since 2006. It is used to provide interoperability between digital content creation applications. FBX is also part of Autodesk Gameware, a series of video game middleware.

Autodesk provides a C++ FBX SDK that can read, write, and convert to/from FBX files.

The FBX file format is proprietary; however, the format description is exposed in the FBX Extensions SDK which provides header files for the FBX readers and writers[6].

2.4.4 DAE (COLLADA)

COLLADA (COLLAborative Design Activity) is an interchange file format for interactive 3D applications. It is managed by the nonprofit technology consortium, the Khronos Group, and has been adopted by ISO as a publicly available specification, ISO/PAS 17506.

COLLADA defines an open standard XML schema for exchanging digital assets among various graphics software applications that might otherwise store their assets in incompatible file formats. COLLADA documents that describe digital assets are XML files, usually identified with a .dae (digital asset exchange) filename extension.[7]

COLLADA has not had an update of its standard since 2008 and is effectively moribund and has been replaced by glTF.

2.4.5 GLTF

glTF (derivative short form of GL Transmission Format) is a file format for 3D scenes and models using the JSON standard. It is an API-neutral runtime asset delivery format developed by the Khronos Group 3D Formats Working Group. It was announced at HTML5DevConf 2016. This format is intended to be an efficient, interoperable format with minimum file size and runtime processing by apps. As such, its creators have described it as the "JPEG of 3D." glTF also defines a common publishing format for 3D content tools and services.[8]

---

[5] https://en.wikipedia.org/wiki/Wavefront_.obj_file
[6] https://en.wikipedia.org/wiki/FBX
[7] https://en.wikipedia.org/wiki/COLLADA
[8] https://en.wikipedia.org/wiki/GlTF

GLTF is not supported by Unity out of the box, however, Khronos-provided SDKs exist to provide design-time and runtime import into Unity. The GLTF standard is open, and the Khronos Group provides open-source implementations for Unity, C#, Blender, and other tools.

2.4.6 BLEND

BLEND is a scene description format associated with the Blender 3d modeling and animation software by the Blender Foundation.

The blend file format is not a true file interchange format, rather it dumps internal data structures from directly from memory to disk. A blend file's structure may therefore be unique to each version of Blender. Despite this, blend files are both backward and forward compatible between versions, and between different hardware and operating systems. This is made possible by the addition of metadata, known as Struct DNA, that allows conversion when loading the file.

Since no standard blend file format specification exists, the Blender source code must serve as the definitive specification.[9]

2.4.7 STL

STL (an abbreviation of "stereolithography") is a file format native to the stereolithography CAD software created by 3D Systems. STL has several backronyms such as "Standard Triangle Language" and "Standard Tessellation Language". This file format is supported by many other software packages; it is widely used for rapid prototyping, 3D printing and computer-aided manufacturing. STL files describe only the surface geometry of a three-dimensional object without any representation of color, texture or other common CAD model attributes. The STL format specifies both ASCII and binary representations. Binary files are more common, since they are more compact.

An STL file describes a raw, unstructured triangulated surface by the unit normal and vertices (ordered by the right-hand rule) of the triangles using a three-dimensional Cartesian coordinate system. In the original specification, all STL coordinates were required to be positive numbers, but this restriction is no longer enforced and negative coordinates are commonly encountered in STL files today. STL files contain no scale information, and the units are arbitrary.[10]

2.4.8 3MF

3D Manufacturing Format or 3MF is an open source file format standard developed and published by the 3MF Consortium.

3MF is an XML-based data format designed for using additive manufacturing, including information about materials, colors, and other information that cannot be represented in the STL format.

As of today, CAD software related companies such as Autodesk, Dassault Systèmes and Netfabb are part of the 3MF Consortium. Other firms in the 3MF Consortium are Microsoft (for operating system and 3D modeling support), SLM and HP, whilst Shapeways are also included to give insight from a 3D Printing background. Other key players in the 3D printing and additive manufacturing business, such as Materialise, 3D Systems, Siemens PLM Software and Stratasys have recently joined the consortium. To

---

[9] http://fileformats.archiveteam.org/wiki/BLEND

[10] https://en.wikipedia.org/wiki/STL_(file_format)

facilitate the adoption, 3MF Consortium has also published a C++ implementation of the 3MF file format.[11]

2.4.9 PLY

PLY is a computer file format known as the Polygon File Format or the Stanford Triangle Format. It was principally designed to store three-dimensional data from 3D scanners. The data storage format supports a relatively simple description of a single object as a list of nominally flat polygons. A variety of properties can be stored, including color and transparency, surface normals, texture coordinates and data confidence values. The format permits one to have different properties for the front and back of a polygon. There are two versions of the file format, one in ASCII, the other in binary.[12]

2.4.10 USD

Universal Scene Description (USD) is the first publicly available software that addresses the need to robustly and scalably interchange and augment arbitrary 3D scenes that may be composed from many elemental assets.

USD provides for interchange of elemental assets (e.g. models) or animations. But unlike other interchange packages, USD also enables assembly and organization of any number of assets into virtual sets, scenes, and shots, transmit them from application to application, and non-destructively edit them (as overrides), with a single, consistent API, in a single scene graph. USD provides a rich toolset for reading, writing, editing, and rapidly previewing 3D geometry and shading. In addition, because USD's core scene graph and "composition engine" are agnostic of 3D, USD can be extended in a maintainable way to encode and compose data in other domains.[13]

USD is forward looking and was announced by Apple as a key component of their ARKit in 2018, but has seen limited uptake by other companies and user communities.

2.4.11 Summary

Augmntr recommends glTF for future work. Its open-source / open-standard nature and wide variety of open-source SDKs outweigh its only negative – the lack of direct support in Unity. This lack of support can be easily worked around though, as Khronos provides both design-time and run-time importers into Unity.

The overall ability to use glTF throughout the Avatar “pipeline” from initial creation to display will allow for flexibility and rigor, and the lack of vendor or platform lock-in will be critical elements for future Phases as hardware and software evolves.

2.5 Pose Estimation / Body Tracking

Pose Estimation / Body tracking is a critical element of MEDIC, as it will generate the 3D points needed to overlay the Avatar over the warfighter in Augmented Reality.

Augmntr reviewed the state of various existing body tracking implementations.

[11] https://en.wikipedia.org/wiki/3D_Manufacturing_Format
[12] https://en.wikipedia.org/wiki/PLY_(file_format)
[13] https://graphics.pixar.com/usd/docs/index.html

2.5.1 OpenPose

OpenPose[14], from the CMU Perceptual Computing Lab, is a software-only library that is the first real-time multi-person system to jointly detect human body, hand, facial, and foot keypoints (in total 135 keypoints) on single images.

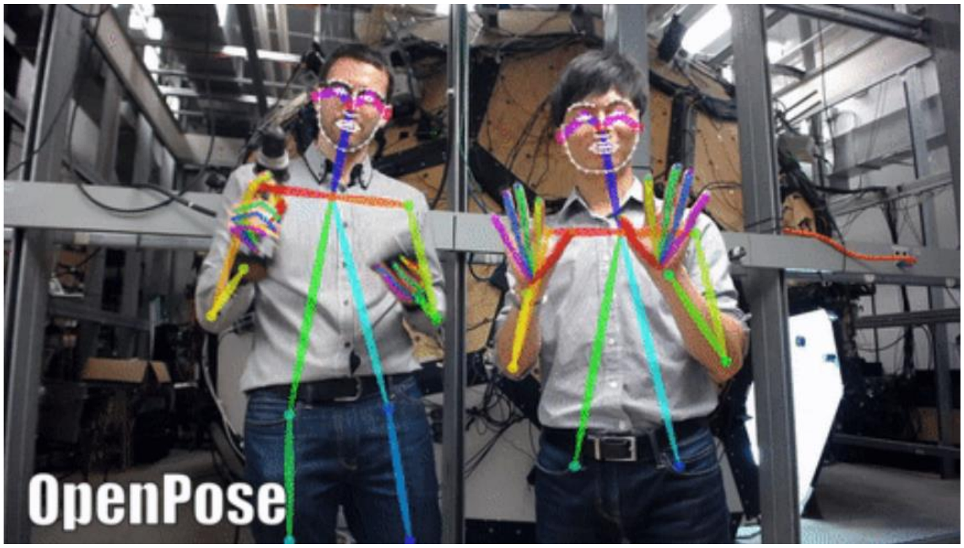


Figure 10: OpenPose Authors Gines Hidalgo (left) and Hanbyul Joo (right) in front of the CMU Panoptic Studio

It is open source for non-commercial purposes, and Commercial Licenses are expensive. It also does not support the use of the depth camera, as present on the future HoloLens devices.

2.5.2 DensePose

DensePose[15], from Facebook Research, aims at mapping all human pixels of an RGB image to the 3D surface of the human body. While powerful, it is only available for non-commercial work Due its lack of support for depth cameras, Augmntr did not enter into negotiations with Facebook for commercial applications.

Figure 11: DensePose results

[14] https://github.com/CMU-Perceptual-Computing-Lab/openpose
[15] https://github.com/facebookresearch/Densepose

2.5.3 NuiTrack

NuiTrack[16] is a company selling 3D Body Skeleton Tracking Middleware. They support multiple types of sensors and run on a variety of platforms. Like many pieces of software, this originally supported earlier versions of Microsoft's Kinect Platform, but has evolved since then into a sensor and platform agnostic piece of software.

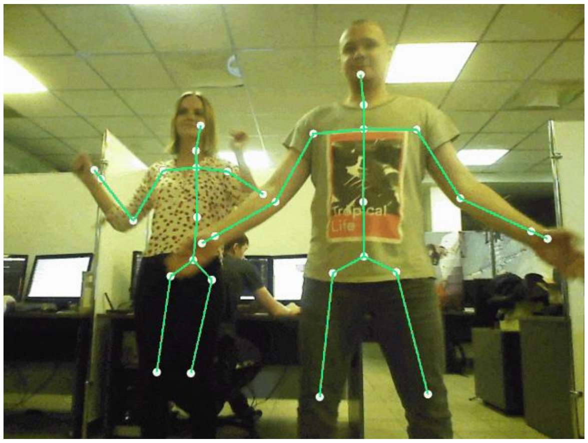


Figure 12: NuiTrack Skeleton Tracking

Figure 13: Orbbec Persee, an example of supported hardware

Augmntr did not utilize this for the initial phase, but due to their support for depth sensors and arbitrary hardware, will consider them for future Phase II work.

2.5.4 Azure Kinect Body Tracking

Microsoft's new Kinect for Azure (K4A)[17] contains the same basic imaging and depth sensors as present in the HoloLens 2.

Figure 14: Kinect for Azure

Figure 15: Depth Sensor common to HoloLens 2 and Kinect for Azure

In addition to the sensor, Microsoft provides the Azure Kinect Body Tracking SDK[18]. This utilizes the depth sensor and is provided in a convenient package.

Augmntr investigated running a Body tracker directly on the HoloLens in Phase I, however this was not possible for several reasons. The HoloLens 1 was a low-powered device and did not have the processing power to run any of the reviewed methods. In addition, low-level access to the depth and other cameras

[16] https://nuitrack.com/
[17] https://azure.microsoft.com/en-us/services/kinect-dk/
[18] https://microsoft.github.io/Azure-Kinect-Body-Tracking/release/0.9.x/index.html

on the device is possible through “Research Mode”, but this can be fiddly and is hard to balance when using the device as intended.

Based on this, Augmntr utilized the external Kinect and a standard Unity application for Phase I, to give the best representation of what the “final” system will be.

Augmntr integrated the Kinect and its Body Tracking SDK with Unity. The Kinect is a separate device for this phase, and so must send its skeleton tracking data over Wi-Fi to the HoloLens. An application was created for an attached PC with an interface allowing the HoloLens to retrieve and decode the raw data via a standard HTTP web service.

The data from the Skeleton SDK provides positions of 26 joints relative to the Kinect, and their rotation. The reference frame convention used by the Kinect is different than that used by Unity, and so had to be adjusted to match.

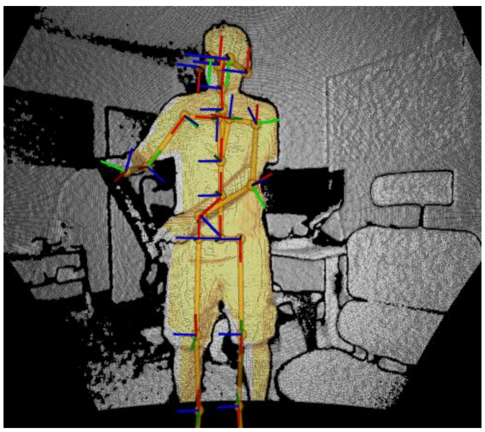
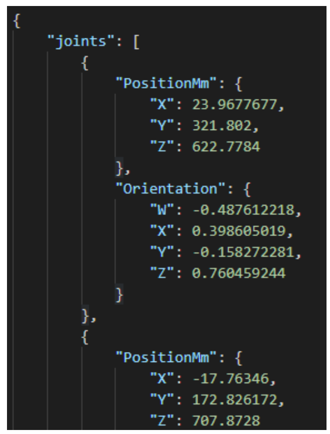


Figure 16: Initial Test from Body Tracking SDK




```
{
    "joints": [
        {
            "PositionMm": {
                "X": 23.9677677,
                "Y": 321.802,
                "Z": 622.7784
            },
            "Orientation": {
                "W": -0.487612218,
                "X": 0.398605019,
                "Y": -0.158272281,
                "Z": 0.760459244
            }
        },
        {
            "PositionMm": {
                "X": -17.76346,
                "Y": 172.826172,
                "Z": 707.8728
```


Figure 17: Output to HoloLens

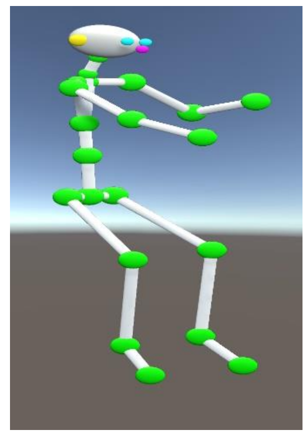


Figure 18: Raw Points in Unity

The Kinect for Azure then had its Vuforia marker tag added so that the two separate reference frames could be combined. One of the “stock” Vuforia markers was used for convenience. This marker is picked up by the HoloLens, and placed in space, and with that, the points from the Azure Body Tracking SDK can be rotated and placed properly into the HoloLens reference frame.

The “rigged avatar” from previous steps was “animated” using the points from the Body Tracking SDK that were transmitted to it over the network. Various optimizations were done on the HoloLens 1 to ensure an acceptable level of performance, and the entire system was tested.

Figure 19: Azure Kinect with added Vuforia Registration Tag

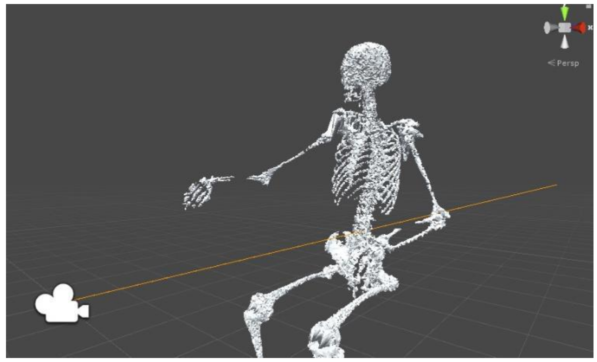


Figure 20: Avatar Rigged and using points from Body Tracking

After the initial development, an update by Microsoft to Body Tracking SDK added "certainty values" to the output for the body tracking points. This tells us which parts the SDK is not sure about, and if so, that can potentially be hidden or displayed in a different way, enabling the user to decide to reposition the camera or be aware that the system is not able to determine body placement.

2.6 Voice Commands

As the MEDIC system is required to be usable in a hands-free mode, much of the system UI must accommodate voice commands. The Microsoft HoloLens and supporting software tools natively support voice commands through the Windows Speech SDK, and this was used for Phase I.

The voice commands were added programmatically. Each organ in the avatar had an autogenerated voice command to "show" and "hide". A few manually created groups were also created to allow for a few groups.

These were tested and functioned as designed, although some more specific groups and specific commands would want to be created with the aid of real-life medics.

A brief list of providers was generated. These providers include:

- Windows built in – option for offline
- Google Speech – apparently online only
- DialogFlow – Built on Google API, online only
- IBM Watson
- CMU Sphinx – offline but needs to be trained manually. Does not come as a 'functional' product
- Nuance Dragon SDK
- Kaldi – runs on a custom cluster

As future Phases will be an off-the-shelf integration activity, the built-in Windows speech recognition, even though it required an on-line connection, was used for Phase I.

2.7 Application Development

Two applications were developed during this Phase I.

2.7.1 Avatar Demo Application

This application, using the 5mm avatar, shows the avatar in full size in front of the HoloLens wearer, and allows the user to slowly spin the avatar, start/stop the avatar from walking, and select different few different views of the Avatar. In this case, the Unity animation system was used to show that the Avatar was being dynamically modified.

Figure 21: Skeleton

Figure 22: Walking Skin

Manual layers of “Skeleton”, “Exterior”, “Digestive”, and “Circulatory” were created, with various organs in each of the groups manually colored. All elements of the avatar were animated.

This sample test application successfully ran at the full 60Hz frame rate of the HoloLens 1.

2.7.2 Full Up Application

The second application integrated the Kinect for Azure (with attached registration target), supporting computer, and HoloLens into a complete prototype application. A camouflage-wearing dummy was built, utilizing the current standard camo pattern, wearing a helmet, and with a 3D printed head. Boxes and gloves were used to provide limbs.

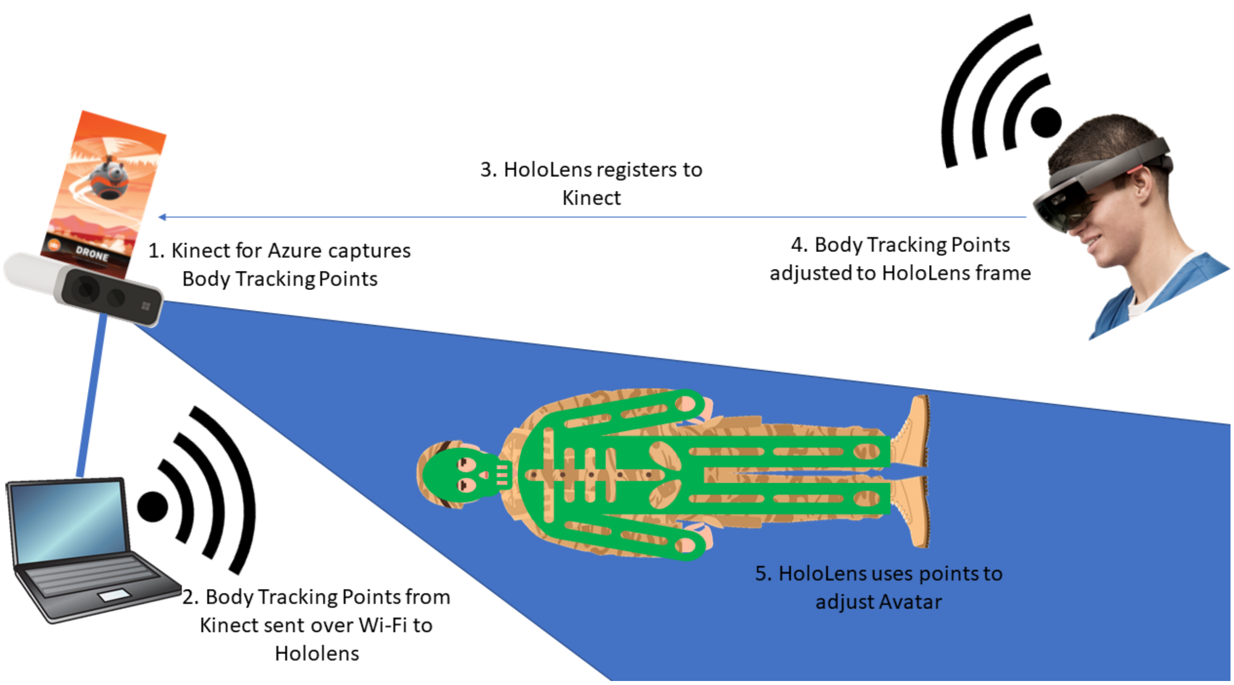


Figure 23: Block Diagram for Phase I prototype

- The registration marker is picked up by the HoloLens, placed in space, and with that, the points from the Azure Body Tracking SDK can be rotated and placed properly into the HoloLens reference frame
- The coordinates of the estimated pose picked up by the Azure Body Tracking are with respect to the Kinect device. The coordinates in HoloLens apps are generated from the point of view when the app starts relative to the location of the device. In order to align the reference points, the tag is pulled into the app as a common point. The coordinates of the body tracking can then be used relative to that tag without having to manually convert the frame as Unity will do that automatically
- The “rigged avatar” was “animated” using the points from the Body Tracking SDK that were transmitted to it over the network. Various optimizations were done on the HoloLens 1 to ensure an acceptable level of performance, and the entire system was tested
- Voice commands were also integrated into the application, allowing easy toggling of the various layers

Combining all these together gives us a prototype application that allows us to overlay the imagery on top of the dummy and switch between the different organs using voice alone.

Figure 24: Outdoor testing with complete prototype system and dummy

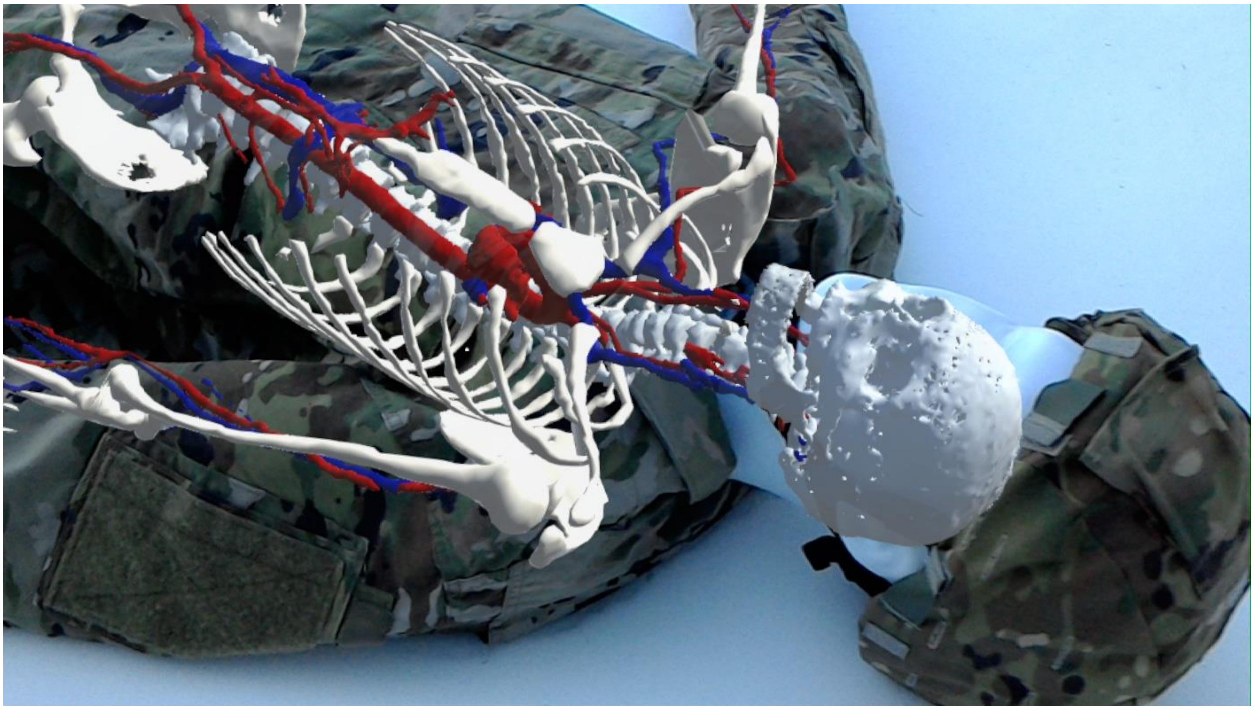


Figure 25: Through-HoloLens image capture of USARIEM Avatar overlaid on the Dummy

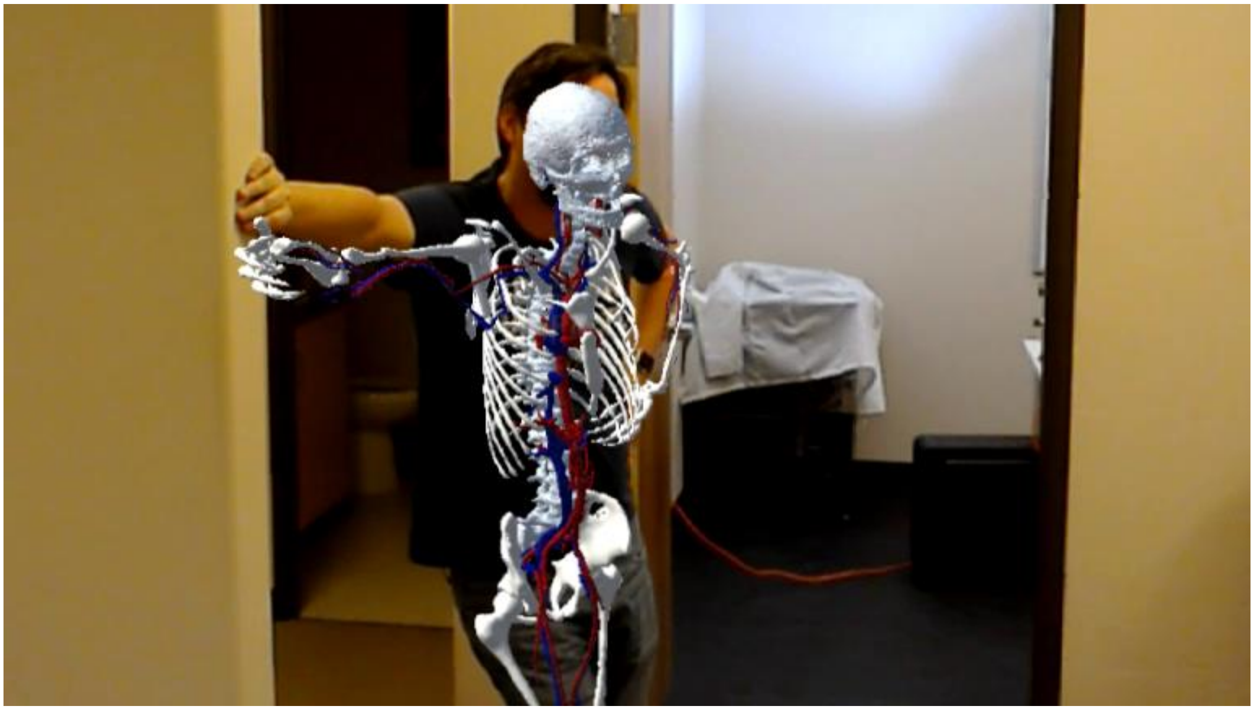


Figure 26: Through HoloLens capture of Initial test with the Avatar overlaid onto the PI in Augmntr's office. Avatar offset due HoloLens camera vs display parallax

2.8 Application Testing

The complete application functioned as designed, allowing a full end-to-end test of the system. This application was tested both by engineering, as well as taken to our medical advisor's office and evaluated by several reviewers, including a trained US Army Reserve 68W Combat Medic. The system was demonstrated, unassisted, by Lafe Redd, Augmntr's CEO, who is not a software engineer and was not directly involved with the technical development of the system.

Figure 27: Prototype being used by Breana Lelak, a US Army Reserve Medic

While not a comprehensive test, the medic, Breana Lelak, had the following comments and questions when using the prototype MEDIC system at Dr. Douglas Redd's Center for Vascular Intervention in Atlanta, GA in late January 2020.

- "Wow. Oh Wow. Dang."
- "I feel like it'd make access points so much easier"
- "So when you put a skeleton up overlaid do you have to have their height and weight?" (Lafe Redd: "No, this automatically scans and adjusts and projects to their height").
- "That is awesome and it'll help with NCDs"

A dozen other employees at the center tried it and were both amazed by AR technology and the HoloLens, but also by the applicability of the MEDIC system to their everyday use.

The following technical notes were made:

- Registration between the camera and headset was rather fiddly. Once lined up, it functioned well, but additional work needs to be performed on really bringing the accuracy in
- The Azure Body Tracking SDK did not functional accurately with a lying dummy. To fix this, the entire sensor was rotated onto its side for this test

- Happily, the Azure Body Tracking SDK did not need all limbs to be present (as this is an unfortunate reality of a medic's job). Limbs that were unable to be found were marked with low-confidence but were shown in our application as-is.
- Separating the Kinect was useful, as it was able to maintain a wide-angle view of the dummy throughout the process. This camera field-of-view vs user field-of-view will be considered carefully for future phases, especially as the cameras on the headset will likely be outside of our control

2.9 HoloLens 2 & IVAS

Since the award of the Phase I, Microsoft and the US Army have progressed on a few fronts.

The HoloLens 2 was announced in March, although as of February 2020 is still in limited allocation. Augmntr has not yet received any (we remain on the waiting list), and so has not been able to benchmark performance. All software developed for this project is easily transferrable to the HoloLens 2, however, and this will likely be the device used for Phase II.

The US Army and Microsoft has also moved out aggressively on the IVAS projects, successfully hitting their first 2 soldier touchpoints. Feedback from soldiers and the progression of the program has been good[19]:

> "When a soldier says, 'this sucks,' it may not be technical, but it has great meaning," said Brig. Gen. Anthony Potts, the Army's modernization director for infantry gear. "We can spend a lot of money building something that meets every single one of our requirements, and if the soldiers don't love it, they won't wear it… and then it'd be a waste."
>
> …
>
> "Our number one factor that we evaluate …with IVAS is this: Do soldiers love it?" he said. "If that's a little bit soft and squishy, I got it — it's soft and squishy."
>
> …
>
> The IVAS effort started out under the nickname "HUD 3.0" in early 2018. "Very few people believed we could build this program in two years," Potts said. "I promise you if we had to stop for everything that I've had to stop for in the past, it would absolutely not be possible. Not possible at all."

It is possible that the IVAS program is consuming a majority of the HoloLens 2 headsets available – pictures online show that the IVAS headset currently appears to be only a slightly modified HoloLens 2.

In addition, it is not clear at this point exactly how other user communities will develop software for IVAS. The Microsoft/US Army team is extremely focused on their program and not stopping for any interaction with other groups at this point. Even key government organizations such as C5ISR Center's Night Vision and Electronic Sensors Directorate, responsible for all night vision devices such as the one on the IVAS headset, had as of the beginning of 2020 received only a few headsets and were "kept under lock and key".

[19] https://breakingdefense.com/2019/12/soldiers-coders-surprise-army-brass-changing-ivas-goggles/

Regardless, as of February 2020:

> The Army also plans on spending $906 million for 40,219 sets of Integrated Visual Augmentation System (IVAS) -- a Microsoft-based system that features sophisticated goggles that allow soldiers to see their weapon sight reticle in their field of view along with other key tactical information. Fielding is set to begin in fiscal 2021[20].

Development of similar mixed commercial/military technology such as the Android Tactical Assault Toolkit (ATAK) utilizes a plug-in model. This allows for the DoD and their selected development partners to easily add additional functionality on top of the core mapping and communication system and have the end users or units cherry-pick the functions they need for their tasks. However, this model, with applications running in 2D on a slightly customized Android smartphone, is not immediately transferable to an AR device such as the HoloLens. Other communities, such as the training, will likely find it harder to integrate into IVAS than a relatively standalone application like MEDIC, but currently questions about sensor integration and the like remain unanswered.

Specifics about application start, load times, and other device-specific features will similarly require deeper interface with the IVAS or equivalent teams as the project progresses. This will be a key focus of our Phase II plans.

3 Key Research Accomplishments

The following are the key research accomplishments accomplished by Augmntr:

- Successful import and display of the government-provided 5mm and 2mm avatars into Unity
- Successful rigging of the Avatars
- Real-time display of a high-quality shaded avatar while maintaining the high framerates required for AR
- Successful integration of an external Body Tracking system into the system, with body points properly referenced to the HoloLens wearer
- Successfully implemented voice commands to control the Avatar
- Successfully created a full end-to-end prototype system that was demonstrated to medical professionals outside Augmntr's development offices

4 Conclusion

The Phase I research effort showed that the entire system concept was feasible and possible on current hardware. The Avatar provided by USARIEM, after some processing and rigging, was capable of being matched up with pose information from a Microsoft Kinect for Azure and displayed in real-time on a Microsoft HoloLens. Speech commands were effective for showing and hiding layers, and the prototype system was demonstratable outside of Augmntr's offices.

The overall system showed great promise, as the comprehensive Avatar data shown in 3D was easy to understand when presented. Even the "generic" Avatar data provided an excellent hands-free reference to the wearer.

---

[20] https://www.military.com/daily-news/2020/02/10/army-pours-cash-long-range-missile-development-new-infantry-rifles.html

Future plans for this project must involve work to make the system field-deployable, fast, and as accurate as it can be.

- Near complete automation of the Avatar rigging and processing pipeline to support custom Avatars
- Deployment of the MEDIC system to IVAS or similar headsets, with any supporting hardware (custom imagers and processing) being extremely light weight
- Improvement in body tracking and avatar adjustment accuracy
- User testing with medics to ensure applicability for the task at hand

In addition to these tasks, Dr Douglas Redd recommended exploring integration of real time ultrasound into the MEDIC system in order to provide more detailed control of the surgical intervention.

Augmntr recommends funding further development under Phase II.

5 Publications, Abstracts, and Presentations

Nothing to Report.

6 Inventions, Patents, and Licenses

Nothing to Report.

7 Reportable Outcomes

Augmntr developed the following key outputs:

- A prototype MEDIC system
- Key understanding of the pipeline needed to turn raw Avatar data into a displayable 3D visualization

8 Other Achievements

Nothing to Report.

9 References

As this was primarily a software engineering development and integration activity, no external journal references were used.

10 Appendices

10.1 Evaluation of MEDIC prototype by Dr. Douglas Redd

Evaluation of Augmtr's MEDIC's IVAS Prototype;

Warfighter medics are challenged by the lack of anatomical imaging data that would normally assist in the treatment of Warfighter injuries. Augmented Reality surgical tool to provide a digitally-superimposed avatar representing the internal anatomy of the Warfighter patient. Warfighter medics using the final MEDIC system will have the opportunity to facilitate better in-field patient care by achieving faster control of exsanguinating internal and external battle field hemorrhage by assisting the warfighter medic to obtain rapid access to the femoral artery or vein allowing in-field insertion of an aortoiliac arterial or iliocaval venous occlusion balloon to control rapid blood loss and stabilize the war fighter by virtue of the MEDIC's digital visualization of the patient's vasculature.

Phase I deliverable report: Key Tasks for successful completion of Phase I are:

1. Develop MEDIC demonstration software – ACCOMPLISHED

Augmntr has successfully demonstrated the capability of their MEDIC's IVAS device to host complex, 3D data sets with preservation of precise temporal and spatial registration achieving precise digital-to-physical registration and precisely aligning and superimposing various digital video layers to the war fighter's digital avatar with precise registration of underlying anatomical/physical structures which will be critical for the success of the MEDIC program.

2. Complete engineering validation - ACCOMPLISHED

Augmntr has successfully validated the engineering requirements of the project by developing a MEDIC IVAS device that utilizes the Warfighter Digital AVATAR provided by the Defense Health Agency. They successfully captured images of a patient using the onboard cameras and using sensors determine the pose of a test patient and demonstrated the ability to track real time movements by the patient while superimposing the Warfighter avatar and its various anatomic layers upon the test patient with the appropriate positioning, scaling and matching the pose of the test patient being viewed through the MEDIC's IVAS headset's display system; and in doing so have executed a functional User Interface that supports all required manipulations, adjustments, display controls and avatar layer selections.

3. Deliver a HoloLens demonstration MEDIC system – ACCOMPLISHED

Augmntr's MEDIC IVAS prototype based upon a commercially available HoloLens system was been delivered for a successful evaluation by Dr. Douglas Redd and several of his staff members at the Center for Vascular Intervention in Atlanta on Thursday, January 30th, 2020.

4. Evaluated demonstration MEDIC system performance – ACCOMPLISHED

Dr. Redd and a number of his staff members, including one of his medical assistants, Breana Lelak, were able to evaluate Augmntr's MEDIC IVAS system at his outpatient vascular surgery center in Atlanta and all of the staff that participated all came away with very favorable opinions about how the application of augmented reality viewed through MEDIC have accomplish all of the tasks that were proposed for Phase 1 of this project. Additionally, it provided Dr. Redd and his staff a glimpse into the future of endovascular surgery that will assist the warfighter medic in the battlefield additional options for performing potentially lifesaving invasive endovascular procedures in the combat theater to control exsanguinating internal or external hemorrhage during the golden hour following a battlefield injury. Not only would this have significant value to the military, but it could also impact the delivery of healthcare throughout the medical field, both in the operating theater, vascular surgery suite or in a clinical setting.

Endovascular surgery is performed by placing a catheter through blood vessels. Due to the fragility of arteries and the difficulty in controlling a long elastic wire to reach the target region, training plays an extremely important role in helping a surgeon acquire the required complex skills. Virtual reality simulators and augmented reality systems have proven to be effective in minimally invasive surgical training. These systems, however, often employ pre-captured or computer-generated medical images as observed in the Warfighter Digital Avatar. However, if such a MEDIC IVAS system were integrated with realtime ultrasound images being captured as the catheter and guidewire were being manipulated, such an augmented reality system for ultrasound-guided endovascular surgical procedures could be superimposed upon and registered with a pre-scanned CT dataset of the patient, or perhaps contained within the warfighters digital dog tag, giving the warfighter medic or the endovascular surgeon much more detailed control over the surgical intervention being performed through augmented reality assisted navigation within the vascular spaces of the body.

10.2 Background on Dr. Douglas Redd

Dr. Douglas Redd, a Vascular Interventional Radiologist and an Atlanta native, is the owner and founder of the Center for Vascular Intervention. Educated at Emory University, he completed his residency in Diagnostic Radiology at Emory and fellowship in Angiography and Interventional Radiology at the University of Pennsylvania.

Influenced by Andreas Gruentzig, the inventor of balloon angioplasty. Dr. Redd spent much of his career in academic, research-based medicine, holding assistant professorships at UPenn and Emory. He left the hospital system in 2005 to establish an outpatient vascular interventional medicine practice and has focused on the minimally invasive treatment of Peripheral Arterial Disease (PAD) for most of the past decade.

The Center for Vascular Intervention was established in response to the epidemic of PAD which is prevalent in metro-Atlanta and North Georgia. By performing highly advanced endovascular procedures below the knee and within the foot, Dr. Redd has become an expert in limb salvage and has prevented numerous patients from undergoing limb amputations. His vision of multi-specialty, holistic care for the treatment of Peripheral Vascular Disease is what makes the Center for Vascular Intervention uniquely positioned to offer patients with the critical limb ischemia complex endovascular treatment options for managing PAD with the goal of amputation prevention.