# U.S. Army Research Laboratory

SUMMER RESEARCH TECHNICAL REPORT

## Investigation of Integrating Three-Dimensional (3-D) Geometry into the Visual Anatomical Injury Descriptor (Visual AID) Using WebGL

AUTUMN KULAGA
MENTOR: PATRICK GILLICH
WARFIGHTER SURVIVABILITY BRANCH, BALLISTICS & NBC DIVISION,
ABERDEEN PROVING GROUND, MARYLAND

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*



| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED |
|---|---|---|
| **AUG 2011** | | **00-00-2011 to 00-00-2011** |

| 4. TITLE AND SUBTITLE | 5a. CONTRACT NUMBER |
|---|---|
| **Investigation Of Integrating Three-Dimensional (3-D) Geometry Into The Visual Anatomical Injury Descriptor (Visual AID) Using WebGL** | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Warfighter Survivability Branch, Ballistics & NBC Division,Aberdeen Proving Ground,MD,21005** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) |
| | 11. SPONSOR/MONITOR'S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT
**Approved for public release; distribution unlimited**

13. SUPPLEMENTARY NOTES
**See also ADA548876**

14. ABSTRACT

**The Visual Anatomical Injury Descriptor (Visual AID) is a graphical computer tool developed to illustrate injury and severity on an anatomical figure, allowing for communication of trauma. Currently, the tool uses two-dimensional images rendered from three-dimensional (3-D) geometry to convey injury. While integrating 3-D geometry was a part of the initial development plan, this step was delayed while waiting for technologies to mature. This paper discusses the Web-based 3-D environment prototype being developed to understand the feasibility of integrating WebGL into Visual AID. Using WebGL will permit us to display the current anatomical geometry used by Visual AID in a 3-D Web-based environment. Employing a 3-D environment allows us to more accurately display the geometry, thus providing a better navigation system and the potential to add annotations fixed to 3-D points. Further developments must be made to better understand integration of highly detailed geometry as well as changing geometry attributes at runtime, such as Abbreviated Injury Scale (AIS) code, severity, and color association. WebGL is a novel cross-platform technology that must be further explored as standardization occurs, thus enabling future advancements. The features that have already been explored will enhance Visual AID?s injury illustrating capabilities, facilitating the communication of injury data.**

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT **unclassified** | b. ABSTRACT **unclassified** | c. THIS PAGE **unclassified** | **Same as Report (SAR)** | **15** | |

## Contents

## List of Figures



## List of Tables

## Abstract

The Visual Anatomical Injury Descriptor (Visual AID) is a graphical computer tool developed to illustrate injury and severity on an anatomical figure, allowing for communication of trauma. Currently, the tool uses two-dimensional images rendered from three-dimensional (3-D) geometry to convey injury. While integrating 3-D geometry was a part of the initial development plan, this step was delayed while waiting for technologies to mature. This paper discusses the Web-based 3-D environment prototype being developed to understand the feasibility of integrating WebGL into Visual AID. Using WebGL will permit us to display the current anatomical geometry used by Visual AID in a 3-D Web-based environment. Employing a 3-D environment allows us to more accurately display the geometry, thus providing a better navigation system and the potential to add annotations fixed to 3-D points. Further developments must be made to better understand integration of highly detailed geometry as well as changing geometry attributes at runtime, such as Abbreviated Injury Scale (AIS) code, severity, and color association. WebGL is a novel cross-platform technology that must be further explored as standardization occurs, thus enabling future advancements. The features that have already been explored will enhance Visual AID's injury illustrating capabilities, facilitating the communication of injury data.

## Acknowledgments

Thanks to Ronald Weaver, Timothy Myers, and Zachary Hamman for their technical insight and programming expertise on this project, and also to Patrick Gillich for excellent mentorship.

## Student Bio

I am a graduate student with a bachelor of fine arts in Communication Arts from Virginia Commonwealth University in Richmond, VA. This undergraduate degree has provided me with the ability to visually communicate information, allowing for a more concise and precise understanding of the information at hand. In order to fulfill the medical illustration track, I completed a concentration in science, focusing on human anatomy. In the fall of 2011, I will be attending the University of Illinois Chicago to obtain a master of science in Biomedical Visualization. This program is one of only five accredited by the Association of Medical Illustrators. For the past five summers, I have been a student employee with the Survivability/Lethality Analysis Directorate in the Warfighter Survivability Branch helping to improve visualization of anatomical geometry.

# 1. Introduction/Background

Visual Anatomical Injury Descriptor (Visual AID) is a graphical tool developed to illustrate injury and severity on an anatomical representation. Illustrations produced by Visual AID visually communicate trauma. Injuries are based on information from casualty medical records or generated from modeling and simulation. Visual AID illustrations allow for communication of injuries to audiences outside the medical community. The first generation of Visual AID uses two-dimensional (2-D) images of prerendered three-dimensional (3-D) geometry (*1*). The development plan of Visual AID included future integration of 3-D rendering within the application. However, a 3-D implementation was delayed while waiting for 3-D Web technology to mature.

Visual AID enables efficient communication of injury data. Injury data is input by manual selection of tissues or using Abbreviated Injury Scale (AIS) (*2*) codes.

AIS is an anatomical scoring system that ranks the severity of injury, providing a consistent technique for describing trauma. AIS is a seven-digit code for unique injury coding. The first six digits detail the location, nature, and type of injury. The seventh, or post dot, digit encodes the severity level. Injury severity is classified according to a six-point ordinal scale with the addition of a seventh value denoting an unknown injury level (table 1). Within the U.S. Army Research Laboratory (ARL), this scale is associated with color values. The associated colors allow Visual AID to display the severity of injured tissues.

Table 1. AIS severity levels.

| AIS | Injury Level | Type of Injury | Example of Injury: HEAD |
|---|---|---|---|
| 1 | Minor | Superficial | Minor laceration of scalp |
| 2 | Moderate | Reversible Injuries; medical attention required | Major laceration of scalp, blood loss < 20% |
| 3 | Serious | Reversible Injuries; hospitalization required | Fracture of skull, penetration < 2 cm |
| 4 | Severe | Non-reversible injuries; not fully recoverable without care | Depressed skull fracture, penetration > 2 cm |
| 5 | Critical | Non-reversible injuries; not fully recoverable even with care | Depressed skull fracture, laceration of spinal artery |
| 6 | Maximal | Nearly Unsurvivable | Massive brain stem crush |
| 9 | Unknown | | Died of head injury without further substantiation of injuries or no autopsy confirmation of specific injuries. |

Injury images are generated using the Zygote Human Anatomy 3-D model (*3*). Use of a reference anatomy independent of personal identification, such as Zygote, allows Visual AID to anonymously display actual patient trauma. The Zygote geometry was broken into components in Autodesk Maya 2009, a 3-D modeling program. Each component was modeled to describe a specific location of injury. Individual tissue components were rendered into 2-D images exported from Maya. Each tissue was rendered in all of the seven colors associated with AIS to

allow for display of severity. All of the tissue images are stored in a directory organized by location on the body and tissue type. A black and white background image was made by rendering geometry from Maya and editing in Adobe Photoshop CS3. When a list of injury codes is input into Visual AID, a stack of prerendered images are displayed on the screen over the prerendered background image (figure 1).

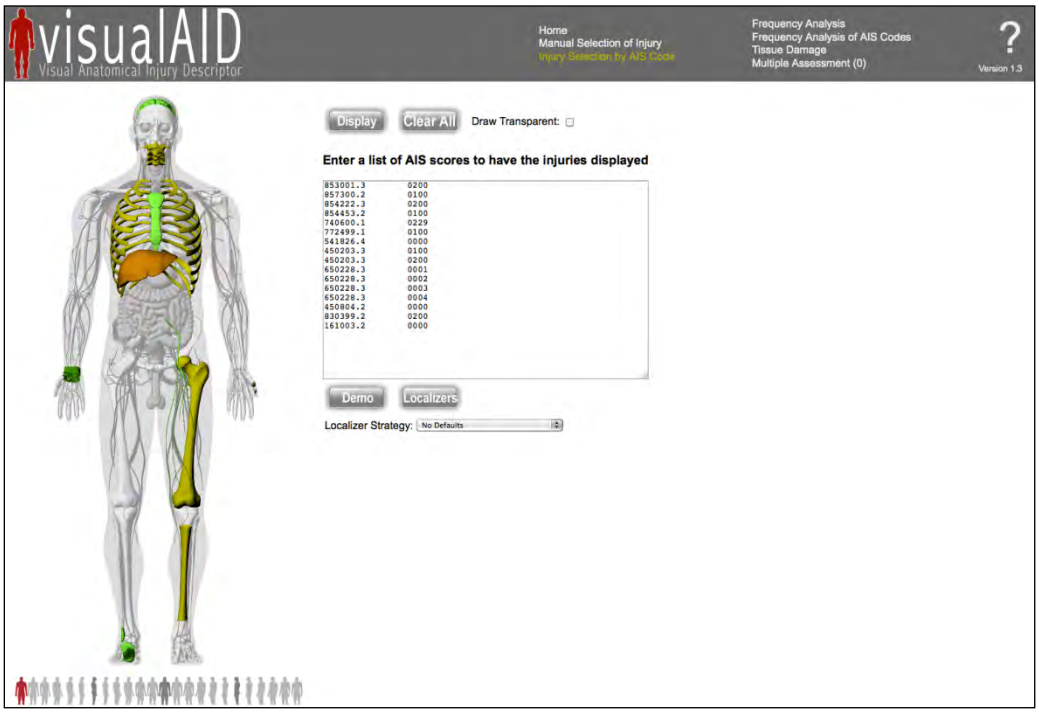


Figure 1. Visual AID user interface.

Recent maturation of 3-D Web technology makes the planned transition from prerendered 2-D images to rendering 3-D geometry within Visual AID a possibility. Web-based Graphics Library (WebGL) is an application programming interface (API) that allows for viewing of 3-D geometry within a Web browser without the need of any plug-ins. WebGL Specification Version 1.0 was released in February 2011 (*4*). This report documents the current effort to understand the advantages, capabilities, and possible costs of integrating 3-D models into Visual AID using WebGL.

## 2. Support for Implementing 3-D

### 2.1 Why 3-D?

The development of Visual AID has allowed for easy production of detailed injury illustrations. Implementation of 3-D geometry allows for more accurate injury illustrations, a better navigation system, and the ability to fix placement of annotations.

When 3-D geometry is prerendered from Maya to 2-D images, the geometry is flattened. When tissues located within close proximity of each other are both displayed, they can produce inappropriate overlap. Visual AID places one image on top of the other based on a predetermined layering order. The current layering order is determined by the distance of the geometry from camera Maya uses to render the image. In figure 2, three images—the lung, the heart, and the coronary arteries, respectively—overlay the background image. The lungs and coronary arteries surround the heart. Portions of the lungs should be hidden behind the heart while other portions of the lungs should overlap the heart. Also, the coronary arteries wrap around the heart. No unambiguous layering is available to portray anatomically correct visualizations. Analysts with less knowledge of anatomy examining the data may confuse and misconstrue the illustrations. In figure 2, arteries displayed could be misinterpreted as existing in front of the heart, or that the lung tissue affected is only behind the heart.

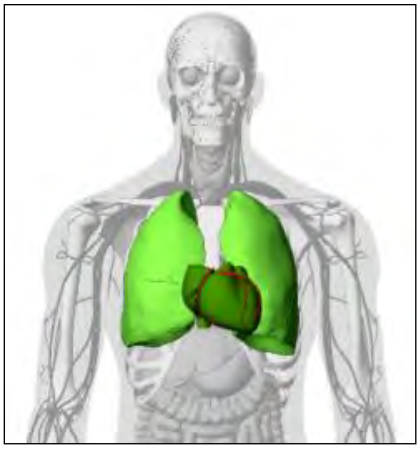

Figure 2. Inappropriate overlap caused by use of 2-D images.

Use of 3-D geometry resolves current issues of overlapping geometry depicted with 2-D images. The Zygote geometry used is defined in 3-D space. If the 3-D geometries representing lungs, heart, and coronary arteries are combined directly in Visual AID, the composite image will display proper interleaving of the tissues. The desire to replace images with 3-D geometry allows Visual AID to provide users with a better understanding of the affected anatomy.

Another advantage to using 3-D rendering is the ability to add a more flexible navigation system. Initial implementation only allowed visualization of injury from the front. The current navigation of Visual AID allows for a view of the anatomy from every 15° along the x-axis. Images are prerendered for each tissue from each supported view angle.

Navigation in 3-D allows for observation from any angle. In addition to rotation along the x-axis, rotation along the y-axis and a zoom option will be added. Adjustment of camera angle can be input to capture any perspective of the injury being illustrated. Once the screen is loaded, the application interchanges injury illustrations to display different angles with little lag.

Furthermore, use of 3-D provides the potential to add annotations that fixed to a specific 3-D point. The current version of Visual AID allows the user to draw a dot in the illustration. The dot is connected to an annotation. If the user decides to view the injury illustration from a different angle, the dot stays in the previous location. The inability to move this annotation along with the geometry is a limitation of Visual AID. This limitation could be fixed in the 2-D version. However, fixing this issue within 2-D implementation would require a rudimentary 3-D context to be developed.

Two functions of the annotation tool are to add notes to specific locations on the illustration and visualizing shot lines. Notes allow for specific comments related to injured tissues. Shot lines are representations of a path taken by a threat, i.e., bullet or fragment. Shot lines are described with the annotation tool by labeling entrance and exit wounds. With the potential to add drawing capabilities in a 3-D context, more mobile annotations can be created.

### 2.2 Why WebGL?

Previously established methods of Web-based 3-D model visualization use plug-ins associated with specific browsers and platforms to add a 3-D context to HTML4. HTML5 adds a canvas tag, which allows for JavaScript embedded in HTML pages to dynamically draw graphics. WebGL, through JavaScript, provides a means to render 3-D geometry in the HTML5 canvas tag. Web browsers with HTML5 and WebGL allow the rendering of 3-D geometry without the use of any plug-ins. The Khronos Group, a nonprofit industry consortium, developed WebGL. WebGL's cross-platform API is derived from Open Graphics Language on Embedded Systems (OpenGL ES) 2.0. OpenGL is a widely adopted 2-D and 3-D graphics API used to communicate with the graphics processing unit (GPU). The use of an already established cross-platform graphics library allows Visual AID to be more adaptable to future developments.

### 2.3 Costs of Implementation

As a new development, WebGL is a maturing technology that has yet to be fully developed and supported. WebGL's lack of full development creates a need for a specific browser, a need to import partitioned geometry, and a lack of documentation.

To use WebGL, a new version of a Web browser is necessary. The Khronos WebGL working group has developed specifications for WebGL version 1.0; however, not all Web browsers support this specification. Mozilla Firefox 4.0 and Google Chrome 10 are the current browsers that support WebGL implementation. Internet Explorer has no plans to develop WebGL capabilities, but a third-party plug-in exists. Also, it is problematic to import very detailed geometry. Detailed geometry equates to more points, vertices, in 3-D space. Vertex indices are used in WebGL to define the points associated with respective polygons. The WebGL working group has placed limitations on the amount of vertex indices associated with one piece of geometry in an effort to create a cross-platform capacity, specifically to support embodied systems such as smart phones. This polygon limitation affects Visual AID by limiting the detail

of tissue geometry or forcing the segmentation of tissue geometry into portions and referencing multiple portions to display a single tissue. The final issue to be discussed is the lack of WebGL documentation needed to allow for efficient learning and implementation. While OpenGL ES 2.0 is the basis for WebGL, and references are available for OpenGL ES, no WebGL-specific books are available. WebGL's current capabilities are tested by referencing online tutorials, wiki pages, and the WebGL Work Group's Specification page (*5, 6*).

As WebGL further matures, the aforementioned developmental deficiencies should be resolved. The WebGL specification is being adopted across browsers on existing and emerging platforms. Also, thorough documentation will follow the expanded support of this novel technology.

## 3. WebGL Prototype and Discussion

To understand the feasibility of integrating WebGL, insight to the WebGL process has been acquired and implementation tests have been performed. In order to provide benefits of 3-D in a Web browser, the abilities that have been explored include understanding the methods of input, assignment of different color shaders to tissues, means to dynamic scene lighting, and control over navigation. To better understand the prospect of integrating WebGL into Visual AID, methods of loading and displaying multiple highly detailed geometries as well as representing separate AIS severities on different tissues must be further explored.

To efficiently run the next version of Visual AID, equipped with WebGL, researchers provided a better understanding of the process WebGL uses to display graphics. JavaScript variables hold buffer and shader information. Buffers hold the data that describe geometry, such as vertex points, vertex indices, and color. A vertex shader and a fragment shader hold different types of shader information, which later define the buffers. Shader programs are built to link variables and shader instructions. These shader instructions are later sent to the GPU along with instructions on how to draw the scene. Knowledge of this process facilitates better development and enhancement of Visual AID.

By following tutorials provided online (*5*), the ability to load geometry and create a dynamic 3-D environment has been prototyped. The development of this project has incorporated heart geometry from Visual AID (figure 3). After understanding the information WebGL needs defined in buffers, such as vertex points, vertex indices, and normal vectors, the information was exported from Maya into text files. The data from the text files were loaded into buffer variables. Once the data-defining geometry is loaded into buffer variables, color can also be defined. Once the GPU has a color defined, the vertex shader facilitates applying the color to the vertex. The fragment shader draws the pixel on the screen and fills in the space between vertices with the correlating color. Once normals are loaded and applied to the vertices, lighting can be integrated into the scene. The normal vectors allow the vertex shader to calculate the direction light reflects off of the geometry, allowing for the appearance of depth of the 3-D shape.

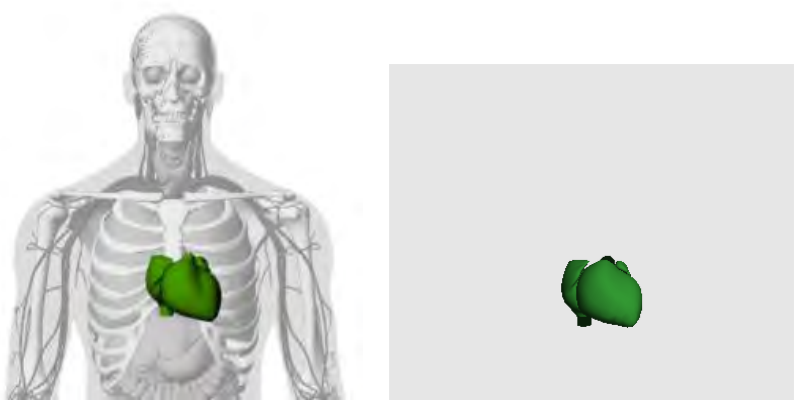

Figure 3. Visual AID interface (left) next to test of Zygote geometry in WebGL environment.

After the tissue geometry was integrated into the scene with color and dynamic lighting, the navigation system was enhanced. A 3-D matrix represents the space where all of the geometry is drawn. In order to adjust the view of the geometry, the matrix is adjusted. To allow for viewing of the heart from multiple angles, the left and right cursor keys adjust the x-axis, the up and down cursor keys adjust the y-axis, and the page-up and page-down keys adjust the zoom view. The geometry data is preloaded into the scene and the view is manipulated, constantly adjusting the position of this data and redrawing the screen.

Development explorations involve fully understanding how to load detailed geometry and how to associate geometry with AIS code to turn on associated color, and methods of adding draw capabilities to add annotations. Currently, it is unknown if latency is an issue when loading all of Visual AID's detailed anatomical geometry. How the application will handle large amounts of data is currently being examined. Geometry with large amounts of indices will have to be reduced or broken into portions and displayed in unison.

Another feature to be evaluated is associating geometry, with correct AIS codes and displaying geometry on the screen. AIS codes can easily be mapped to array variables defining tissue data. Localizers are a numeric code in AIS that further specifies location of an injury. Given the common use of localizers when analysts code AIS, we need to determine if association of only portions of tissue data can be mapped to AIS codes without duplicating entire data files. The benefits or costs of not duplicating this data must be further understood.

Finally, the best method to load geometry and control visibility at run time needs to be better understood. There is the potential to have differences in application performance with preloaded geometry vs. geometry loaded while running the application. The GPU runs the shader programs once when the page is initially loaded; this is the point when the geometry is loaded onto the scene. The current options to explore include loading the geometry at once and adjusting visibility, or reloading the scene to adjust displayed tissue data and AIS severity. The most efficient way to perform such operations will fully be understood once large amounts of data can be properly loaded into the scene.

Current external efforts on the Web, such as Google Labs (*7*), support the prospect of integrating WebGL into Visual AID. In this prototype, benefits of 3-D have been realized by understanding the methods of input, assigning different color shaders to tissues, and adding dynamic scene lighting and navigation. To provide an injury illustration tool with better capabilities and control over the environment, methods of loading and displaying multiple highly detailed geometries, as well as representing separate AIS severities on different tissues, must be better understood and implemented.

## 4. Summary and Conclusions

As a computer graphics tool, Visual AID allows for effective communication of injury through descriptive illustrations. The current method to display injury or trauma data is by displaying 2-D images of 3-D geometry. Implementation of a 3-D environment will facilitate Visual AID's ability to provide more accurate injury illustrations, navigate fully around the geometry, and more accurately place annotations. These implementations will allow for more effective representations of trauma. WebGL is a viable means to implement the 3-D environment given its cross-platform/cross-browser abilities and lack of need for Web browser plug-ins. The full support of WebGL by Web browsers is still under development; however, multiple sources (*4–6*) are available to support integration of this technology into Visual AID.

Understanding the prospect of adding 3-D into the Web environment was done by integrating WebGL to control the data input, assigning different color shaders, and adjusting dynamic lighting, as well as improving navigation. As further investigation and creation of a WebGL prototype continues, the methods of loading multiple highly detailed geometry and assignment of AIS code with severity to tissue data will be examined. With the enhancement of Visual AID through integration of a 3-D environment, more accurate and descriptive injury illustrations can be made. The use of an established cross-platform graphics library, such as WebGL, will facilitate many of Visual AID's future advancements.

## 5. References


1. Kulaga, A. R.; Gillich, P. J. *Visual Anatomical Injury Descriptor (AID)*; Survivability/Lethality Analysis Directorate, U.S. Army Research Laboratory, July 16, 2010.
2. International Injury Scaling Committee. *Abbreviated Injury Scale (2005 Revision)*; Association for the Advancement of Automotive Medicine: Arlington Heights, IL, 2008.
3. Zygote Media Group Web site. Zygote Human Anatomy 3D Model, 2010. http://www.zygote.com/ (accessed July 26, 2011).
4. Khronos Group Web site. Khronos to Create New Open Standard for Advanced Device and Sensor Input, April 12, 2011. http://www.khronos.org/news/press/releases/khronos-webgl-initiative-hardware-accelerated-3d-graphics-internet (accessed 26 July 2011).
5. Learning WebGL Web site. http://learningwebgl.com (accessed 26 July 2011).
6. Khronos Group Web site. WebGL Specification, 10 February 2011. https://www.khronos.org/registry/webgl/specs/1.0/ (accessed 26 July 2011).
7. Google Body Web site, 2011. http://bodybrowser.googlelabs.com/ (accessed July 26, 2011).