LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-CONF-530451

Visualization of Target Inspection data at the National Ignition Facility

D. Potter, N. Antipa

February 21, 2012

IAEA 8thTechnical Meeting
San Francisco, CA, United States
June 20, 2011 through June 24, 2011

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Visualization of Target Inspection data at the National Ignition Facility

Daniel Potter, Nick Antipa
Lawrence Livermore National Laboratory

As the National Ignition Facility continues its campaign to achieve ignition, new methods and tools will be required to measure the quality of the target capsules used to achieve this goal. Techniques have been developed to measure capsule surface features using a phase-shifting diffraction interferometer and Leica Microsystems confocal microscope. These instruments produce multi-gigabyte datasets which consist of tens to hundreds of files. Existing software can handle viewing a small subset of an entire dataset, but none can view a dataset in its entirety. Additionally, without an established mode of transport that keeps the target capsules properly aligned throughout the assembly process, a means of aligning the two dataset coordinate systems is needed. The goal of this project is to develop web based software utilizing WebGL which will provide high level overview visualization of an entire dataset, with the capability to retrieve finer details on demand, in addition to facilitating alignment of multiple datasets with one another based on common features that have been visually identified by users of the system.

Keywords: NIF, Visualization, WebGL, Web Application

1. Introduction

The National Ignition Facility (NIF) [1] at Lawrence Livermore National Laboratory is the world's largest laser. It has been built with the goal of being the first facility to demonstrate controlled laser-driven ignition. This is achieved through a process called inertial confinement fusion (ICF), whereby extreme force is applied to a hollow spherical, BB-sized capsule containing fusion fuel, causing the fuel to compress and heat until a fusion reaction occurs. Due to the extreme temperature and pressure required to achieve ignition, energy must be distributed very uniformly around the capsule during implosion. Isolated features on the surface of the capsule can cause an uneven implosion, leading to radiative cooling of the implosion core and lowering the probability of achieving ignition.

To better evaluate the target manufacturing process, and to gain insight on how the shape of capsule surfaces affect the performance of NIF experiments, each spherical target capsule's surface is measured twice: first using a phase-shifting diffraction interferometer (PSDI) [2] then, later in the assembly process, a Leica Microsystems "Leica DCM-3D", a surface profiling programmable array confocal microscope. These instruments each produce a set of images, containing both topography and reflectivity information, that cover the entire capsule surface. A single dataset can produce up to 1000 images, totaling several gigabytes in size. These images can be viewed individually, but a method of viewing the complete 3D dataset in its spherical geometry is desired.

This paper introduces an image based visualization system for data exploration of target shells at the NIF. The visualization software combines multiple image sets into a single visualization in order to facilitate alignment of data sets, and to provide a method of navigating the data in ways that are not possible with existing tools.

The next section describes the design of the overall system, from the acquisition and storage of this data, to its visualization.

3. System Design

Software has been developed to visualize this data in a three dimensional space, much like Google Earth can be used to view geographical data. Fig. 1 shows the system architecture of the application.

3.1 Inspection and data storage

Raw image data and metadata are collected from the microscopes and uploaded to an Oracle database. For the Leica microscope data, an image analysis program is run which identifies features of interest in the individual images and records their locations. During the imaging process for both systems, the capsule is mounted to a rotating stage which allows the entire surface to be imaged. The theta/phi position of the rotating stage at the time an image is taken is stored as metadata with every image, as well as other important metadata such as pixel and image size. Display images are created from the raw image data during the upload phase. Images are stored at multiple resolutions so that lower resolution images can be downloaded first in the visualization, and higher resolution images can be downloaded on demand. This reduces the initial wait time for the user. These display images are stored in the PNG image format, and add less than 10% to the storage requirements of the raw data.

author's email: potter15@llnl.gov, antipa1@llnl.gov

Prepared by LLNL under Contract DE-AC52-07NA27344.

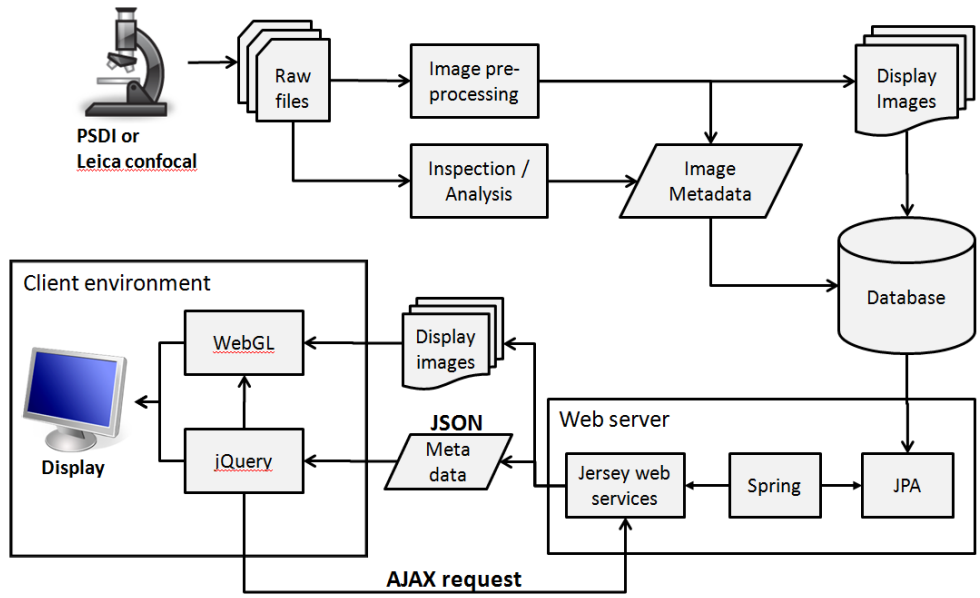

Fig.1 Raw microscope image data is picked up by image processing routines which analyze and pre-process the images for visualization. The visualization layer is implemented as a web application.

3.2 Web application architecture

Our web application relies on several open source technologies and frameworks. On the database and server side framework end we use JPA and Spring. JPA, or the Java Persistence API, is a framework which allows us to easily map database objects to java classes, simplifying communication with the database. We use the Spring framework to create and inject these data access objects into the application during initialization of the application server.

On the client side we build our user interface using jQuery, and communicate with the server side Java code and JPA data objects using RESTful web services. jQuery is a library built on top of Javascript which simplifies development of Javascript code and solves many cross browser compatibility problems associated with Javascript. We have found that jQuery facilitates a separation of concerns with different modules of the UI with its simplified event model and extended libraries such as JavascriptMVC, which add some object oriented capabilities to javascript programs. jQuery also allows us to easily make AJAX (i.e. XMLHttpRequests) to provide a rich internet application with quick response times.

Our AJAX requests are made against RESTful web services, web services made accessible over a URL using the HTTP protocol. These serve as a data source that can be directly and easily accessed through jQuery's AJAX interface. We use the Java based Jersey API for our web services. We find that a simple service can be set up in a minimal amount of time using this framework. The Jersey API also has plug-ins that can automatically convert java objects into JSON data formats. JSON, or JavaScript Object Notation, is a format similar to XML but tends to be more concise. It is also a convenient format to use in Javascript based applications.

Converting our database model objects into JSON enables us to use them in jQuery/Javascript in the same way we would use them in server side java code. This simplifies the design of the system as the representation of objects is consistent at all layers of the application. 3D rendering is achieved using the WebGL framework, while all other UI elements and logic is handled using jQuery based libraries.

3.3 Functionality of the application

The functionality of the application is inspired by Google Earth, where users can rotate, zoom, and pick out sites of interest on an interactive 3d visualization of the Earth. Each image is positioned in a 3D space, creating a spherical surface that the user can interactively rotate and zoom in on.

To position images, it first queries the radius of the target, the theta/phi coordinates of the stage relative to the microscope and pixel width/height of each image belonging to the data set. A geometry patch is generated representing the surface curvature spanning the image, and each image is texture mapped onto its corresponding geometry patch. Simple quads were originally used, but the curved geometry makes some overlapping image features line up better, and makes the overall target look more spherical when completely zoomed out. When the image is first loaded, its spherical coordinate values are converted to Cartesian coordinate space, this is where the geometry is placed during rendering. Before rendering occurs, each geometry patch is rotated about its local Z axis by the images phi value, and rotated about its local X axis by

author's email: potter15@llnl.gov, antipa1@llnl.gov

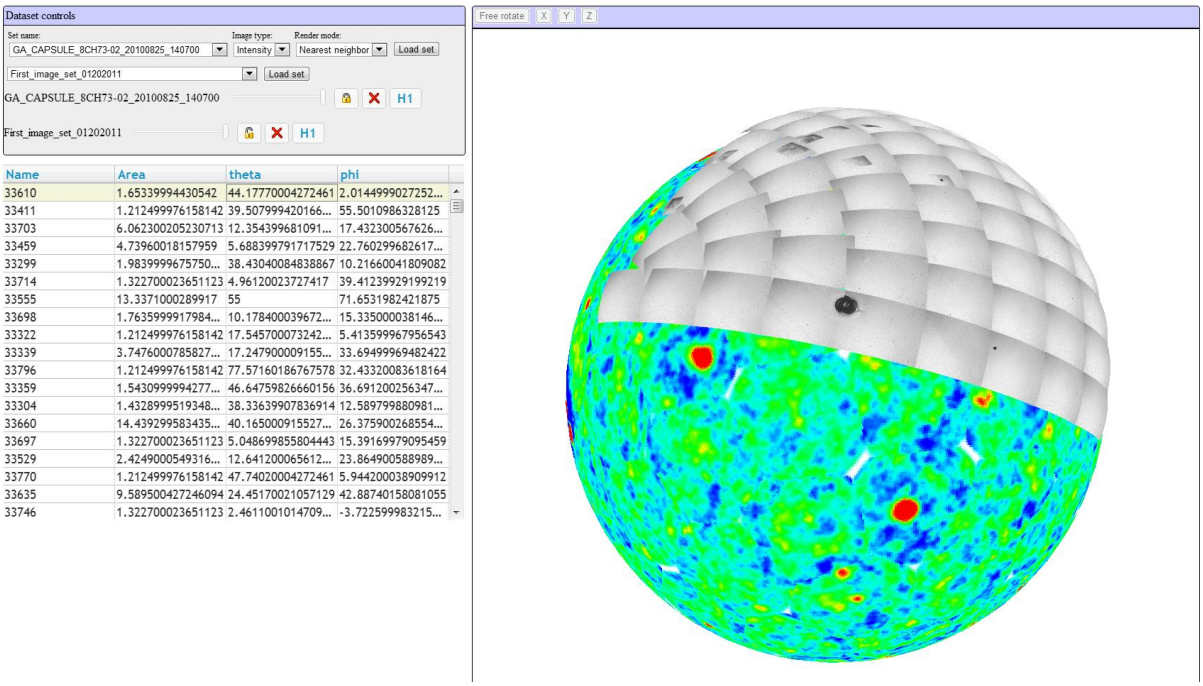

Fig.2 View of the application. Image controls are shown in the upper left, sites of interest are listed in the lower left. The 3D visualization can be interacted with in the box on the right side. Here two datasets are loaded, one covering an entire capsule and the other covering 1/8th of the capsule.

the images theta value, making it face outwards at the proper angle. The final result is displayed in Figure 2.

Figure 2 should give an idea of the utility of this visualization due to the fact that over 200 images are involved. The darker borders of the images in the smaller grayscale dataset make their size obvious. Images in the other dataset are roughly the same size. It would clearly be much harder to view these datasets image by image, or even in groups of images. An advantage to being able to view the entire dataset at once is that it gives users a spatial context for each individual image.

Another feature provided is the listing of features of interest found during the inspection phase mentioned in section 3.1. Users are able to sort this list by any attribute of interest. Upon clicking a row in the list, the view automatically rotates to the location of the feature that was clicked. This is done by calculating the angle between the viewing vector and the location vector of the feature, then rotating by that angle about a rotation vector parallel to the plane defined by the viewing vector and location vector.

Data set alignment is another task that would be very difficult to perform if viewing data image by image. As previously mentioned, the visualization currently supports datasets from two types of microscopes. We currently do not have a way of keeping the target orientation consistent between workstations, so features will not automatically line up between datasets. This visualization provides an easy way to find common features visually. First, the user can lock a dataset from being rotated, allowing them to line up features in the other set. To assist with this process, a slider is provided to control the transparency of each dataset. Second, the user can define an axis of rotation anywhere on the sphere, which locks rotation of the set about that axis. When the user lines up one common feature, they define an axis of rotation in the center of the feature. They then lock one of the sets, and rotate the other about the fixed axis until the rest of the common features line up. This alignment is then saved to the database, so future viewing of the datasets will already be aligned when visualized.

4. Discussion – Why a web application?

The user interface was implemented as a web based application because the centralized nature of a web application makes them easier to develop, test, deploy, and support. However until recently, the sort of 3D visualization we wanted for this application would not have been desirable as a web application. The current state of 3D content on the web is summarized in [3]. Previous web based 3D frameworks had drawbacks such as not reaching maturity, using proprietary programming languages, or being inconvenient to use, which prevented these frameworks from gaining widespread use.

A new technology has recently been incorporated into modern browsers called WebGL. This framework allows rendering of 3D content directly in the browser

author's email: potter15@llnl.gov, antipa1@llnl.gov

without the need of any plug-ins. The user does not need to perform any setup in order for it to work as long as they are using a supported browser. As of this writing supported browsers are Chrome and Firefox 4, with support in upcoming versions of Safari coming. WebGL is a promising framework for the future of web based 3D content. First WebGL utilizes local graphics hardware to deliver performance that most other web-based 3D frameworks cannot match. This also gives developers access to run shader programs directly on the GPU, allowing them to implement modern GPU based algorithms. Second, while it is a new specification for web browsers, it is based on the well known and widely used OpenGL framework. The API is based off of OpenGL ES 2.0, a subset of the OpenGL 2.0 library, used primarily for 3D rendering in mobile devices such as the iPhone and Android platforms. WebGL also has an active community, with many budding new frameworks being created to ease its use such as [4-6] and research using it for web based visualizations [7].

5. Conclusions and future work

This application has been used by various groups at NIF that deal with the imaging of target surfaces. Despite being in its early stages it has already proven to be a valuable tool for exploration of these datasets. In some cases it has provided insight that could not have been practically gained by viewing data image by image.

The primary system requirement for a web based 3D application such as this is the local graphics hardware. Because this application loads a large amount of images onto the video card, a fair amount of video memory is required. Currently the application will run well on mid range graphics hardware with 512MB of memory.

Future work on the application will involve adding new features to aid in aligning datasets, such as adding a basic image annotation system, as well as giving the user more options in how individual datasets can be rotated and positioned. Performance will also be addressed to target low end graphics hardware with 256 MB of video memory, in order to expand the number of supported platforms. Optimizations have been made to achieve this such as loading lower resolution images by default, and only loading higher resolution images if they contain a feature that is of interest. This could be further optimized by dividing each image into smaller sections, and only loading the high resolution version of the smaller section that contains the feature of interest.

Acknowledgements

We would like to thank Laura Kegelmeyer for bringing the necessary people together in order to work on this project and Michael A. Johnson for his help throughout the project.

References

[1] National Ignition Facility Programs. https://lasers.llnl.gov/

[2] R. Montesanti, M. Johnson, E. Mapoles, D. Atkinson, J. Hughes, J. Reynolds. Phase-Shifting Diffraction Interferometer for Inspecting NIF Ignition-Target Shells. Lawrence Livermore National Laboratory UCRL-PROC-225005.

[3] S. Ortiz Jr. Is 3D Finally Ready for the Web? (Technology News, January, 2010) p. 14.

[4] M. Benedetto, F. Ponchio, F. Ganovelli, R. Scopigno. SpiderGL: a JavaScript 3D graphics library for next-generation WWW (Proceedings of the 15th International Conference on Web 3D Technology, Web3D '10, ACM, New York, NY, USA, 2010) p. 165-174.

[5] B. DeLillo. WebGLU development library for WebGL (ACM SIGGRAPH 2010 Posters, SIGGRAPH '10. ACM, New York, NY, USA, Article 135, 2010) 1 pages.

[6] C. Leung, A. Salga. Enabling WebGL (Proceedings of the 19th international conference on World wide web ,WWW '10, ACM, New York, NY, USA) p. 1369-1370.

[7] M. Callieri, R. Andrei, M. Benedetto, M. Zopp, R. Scopigno. Visualization methods for molecular studies on the web platform (Proceedings of the 15th International Conference on Web 3D Technology, Web3D '10, ACM, New York, NY, USA, 2010) p. 117-126.

author's email: potter15@llnl.gov, antipa1@llnl.gov