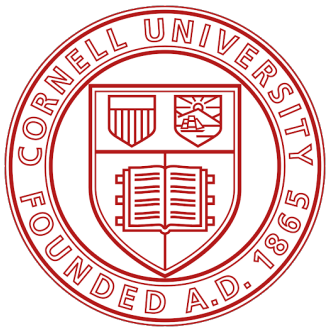

Cornell University
Cornell High Energy
Synchrotron Source
Ithaca, NY

Fort Lewis College
Department of
Physics and Engineering
Durango, CO

Visualizing 3-D Residual Strain Maps from Energy Dispersive Diffraction Data on CAD Models Using ParaView

Authors:
Wiley Kirks

Advisors:
Kelly Nygren
Chris Budrow

Date: August 3, 2020
Summer 2020

Abstract

The goal of the project was to display and visualize residual strain maps measured using energy disperse diffraction in a metallic component using the software ParaView, allowing clear communication of synchrotron data and results with CHESS collaborators. Throughout the research process several data file formats were identified and analyzed for compatibility with ParaView. Python code based programs including JupyterNotebook and Spyder, along with pre-existing functions and external libraries, were used to write the beamline data into readable files. Visualization Tool Kit (VTK) files were found to be the most efficient and useful files to represent data within ParaView. The VTK files allowed for the storage of structural data of the component along with residual strain data observed with in the part. The ParaView 'Delaunay' data filter allowed for easy strain data interpolation of the part. The new visualization framework is demonstrated for a residual strain map collected beneath a weld in a steel plate.

1 Introduction

In science and engineering, it is common for researchers to work with external collaborators who may not share a common knowledge base or be familiar with their techniques. Communicating the data and results to these collaborators is essential for the success of the project and/or research. At the structural materials beamline (SMB) at the Cornell High Energy Synchrotron Source (CHESS), beamline scientists analyze the residual strain with in metallic components through energy dispersive diffraction (EDD). The strain data helps to understand the structure of the material and its response to thermal and mechanical stresses. This knowledge can hopefully help to avoid future structural failures of the component. SMB collaborators are not expected to be (or necessarily become) X-ray experts, and the data is not at all intuitive, so communicating the strain data effectively is imperative. The goal of this research project is to display and visualize the residual strain of any structural component through the use of 3-D ParaView CAD models.

1.1 Energy Dispersive Diffraction at SMB

EDD is a high-throughput diffraction technique that is sensitive to small changes in the crystal lattice that can be attributed to elastic strains in a volume of the material. The technique at SMB utilizes a polychromatic X-ray beam with an energy spectrum of 50-200 keV. These high energies can penetrate thick metallic alloys (up to 1" of steel). Currently, the technique uses one energy-discerning detector downstream of the sample at a fixed angle. A set of beam-defining slits before the sample, and two sets of collimating slits downstream of the sample on the fixed angle of the detector arm define a diffraction volume that remains fixed in space. The diffraction pattern from this region will have discrete peaks at different energies which can be fitted to extract an elastic strain. The sample

can be rasterized in the fixed diffraction volume to create a strain map. The sample is oriented to target specific components of strain.

1.2 Data Process Overview

The EDD measurements were taken at the SMB then converted into strain measurements using a Jupyter Notebook python script previously written by one of the beamline scientists. Python, Spyder or Jupyter Notebook was then used to write the data into a readable file that was compatible with ParaView. ParaView is able to read many different files and viewing the file is then simply done by uploading it. ParaView offers several data analysis filters that allow the user to view and manipulate the data. Figure 1 displays a flowchart that shows the complete progression of the experiment, from the beginning X-ray techniques used to the final ParaView model. Figure 1 also goes into more depth on the process of displaying the strain measurements as data in ParaView.

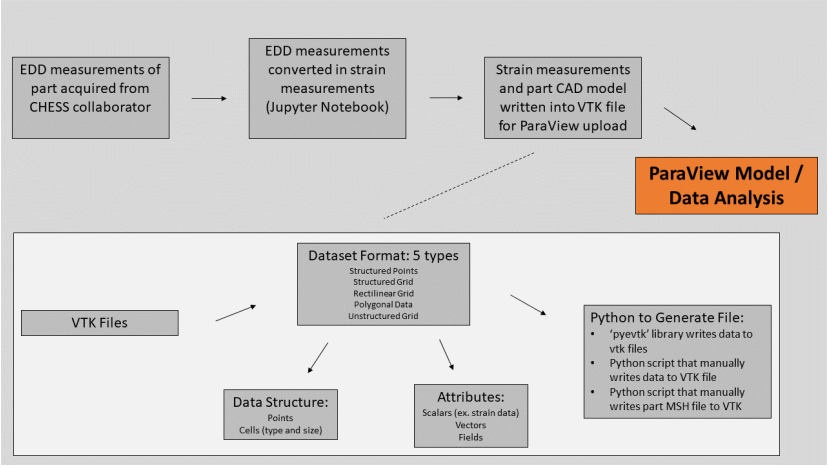

Figure 1: Flowchart of strain measurements to visualization in ParaView.

1.3 Part and Data Background

The data used during this research, and referred to in this report, comprise 2-D strain measurements of a welded structure acquired from the Naval Surface Warfare Center (NSWC). CHESS SMB scientists worked with engineers from the NSWC to analyze the strain effects of thermal and mechanical stresses after undergoing welding procedures. Figure 2 displays a picture of the actual component that was studied and analyzed. The EDD strain measurements were also compared to a finite element simulation of the welds. The strain data that will later be rendered using ParaView is a map of the 'xx' component of strain beneath a 4-pass weld, the location of which is shown in Figure 2.

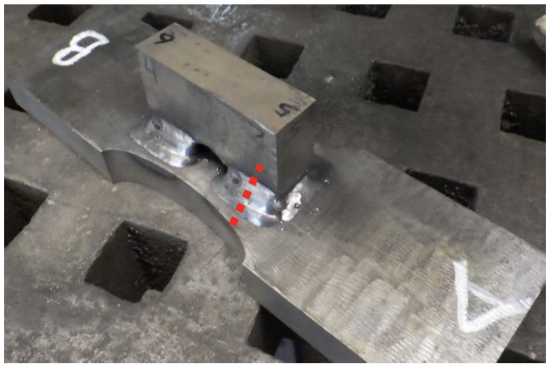

Figure 2: Component acquired from NSWC and analyzed at SMB using EDD. The red-dotted line represents the section of weld that was analyzed.

2 Data Files

ParaView is capable of reading in many different types of data files, some though are more reasonable and useful than others. The spatially-resolved strain data needed to be written into one of the readable files to be viewed in ParaView. Throughout my project and research, VTK files were found to be the most useful and practical. STEP and MSH files were required for writing CAD models of the component into VTK files. Although not as versatile, TXT and CSV files were found useful in some instances.

2.1 VTK

Visualization Toolkit files (VTK) are the primary files used for data analysis in ParaView. Like ParaView, the Visualization Toolkit is an open source platform provided by the company Kitware, so the two are very compatible with one another. VTK files are useful for containing several sets of data regarding a model in one file. The purpose of VTK files is to contain not only the structural data such as the points and cells of a model, but other desired scalar or vector data pertaining to individual points or cells as well. VTK files can take on 5 main types of data set formats: structured points, structured grid, rectilinear, polygonal and unstructured grid data [1]. The 2-D strain data that was analyzed during this project was written into a 'VTK points' data set before being interpolated in ParaView. The following PDF "VTK File Formats: For VTK Version 4.2" is an excellent resource for information on the structure of these files and how to build them [1].

2.2 TXT/CSV

Using Python the strain data could be written into TXT or CSV table data. ParaView can read in these files very easy due to there simple format. These

files are not as complicated or large as the VTK files so they lack in some areas but do have their strengths. These types of files were quite useful for reading simple 2-D strain data into ParaView without the need of a large robust file like a VTK. With that, please note that they do not have the data structure capabilities of the VTK files which are useful for complicated data and models.

2.3 STEP/MSH

It was found that IGES finite element files were very cumbersome to work with, especially in the SolidWorks platform as well as ParaView. To solve this issue, STEP and MSH files were used in the python process to read CAD models such as SolidWorks models into VTK files. CAD models in SolidWorks were exported as STEP files which were then converted into MSH files using the finite element platform 'GMSH'. The following MSH files were then written into VTK files using python and thus could be uploaded into ParaView.

3 Python

Python was the only and primary code used for writing the required scripts that would write the strain data into different data files. Spyder and Jupyter Notebook were the two interfaces used in this process.

3.1 Writing VTK Files

Two primary methods were used for writing the strain data into readable VTK files. The first was an external python library called 'pyevtk' [2]. This library allows the user to write data into a VTK file of any of the desired data formats. It also allowed for the addition of multiple scalar values to be assigned to the data points and cells. The 2-D strain data analyzed in during this research was written as point data and assigned a scalar value 'Strain' for each data point in space. The use of this library requires Numpy 1.11.3 and is compatible with Python 2 and 3. The CAD MSH files were read and written into a VTK using a Python function previously written by another beamline scientist.

3.2 ParaView Python Based

ParaView is a Python based platform so there are a few things worth mentioning with regards to this. ParaView can be completely run through a python command prompt in the ParaView GUI. This prompt can be enabled by selecting the 'View' tab and then selecting 'Python Shell'. Since ParaView is Python based, anything created from the ParaView GUI can be reproduced using Python code. A valuable tool worth noting is the 'Trace' tool. By selecting the 'Start Trace' function in the 'Tool' tab any following commands or processes that happen within the Paraview pipeline are recorded. Once complete, selecting 'Stop Trace', ParaView provides a complete Python script that shows every

step that created the data, part, model during the Trace.[3] This script can then be run from a Python environment.

4 ParaView

4.1 Filters

ParaView is a data analysis platform that utilizes 'Filter's' to display, analyze, and manipulate the data being viewed. An excellent resource for ParaView how to and usage is "The ParaView Tutorial: Version 5.6" [3].

4.1.1 TableToPoints

The 'TableToPoints' filter was very useful for when TXT/CSV table data was uploaded in to ParaView. The filter converts selected columns from the data table file to be used as 2-D or 3-D points. Any column of available data from the table can be used to visual color our gradient fields throughout the point data. Figure 3 shows the use of the 'TableToPoint' filter on 2-D strain data taken from a component at the beamline. The data was then colored according to the strain value of each individual point. The scale coloring represents negative (compression) and positive (tension) strain, with red and blue displaying the positive and negative strain, respectively.

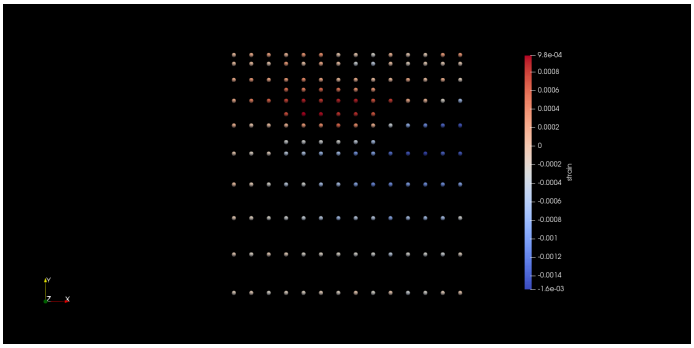

Figure 3: 2-D individual point strain data represented using 'TableToPoints' filter in ParaView.

4.1.2 Delauney 2D/3D

The 'Delaunay 2D' and 'Delaunay 3D' filters allowed for the triangulation of the point data so as to interpolate the strain data across cells. The only difference between the two different Delaunay filters is the desired dimension of the data being analyzed. In Figure 4 the 'Delaunay 2D' filter was used to triangulate and interpret the 2-D point strain data from Figure 1. This allowed for a much clearer understanding of the residual strain acting throughout the component.

The 3-D filter will actually interpolate in 3 dimensions and with the 'Clip' filter the user can analyze internal data of the component. This filter can be seen with the final part rendering of the component and strain data.

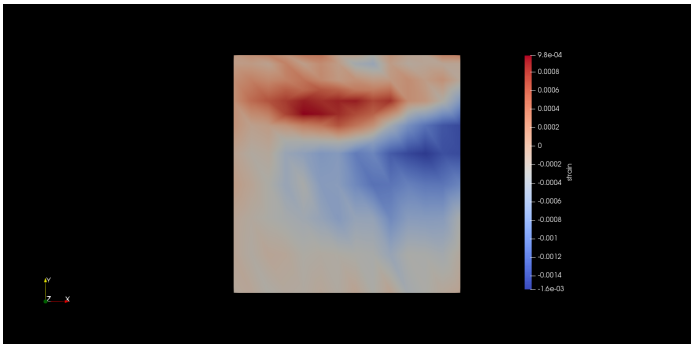

Figure 4: Delauney 2-D triangulation of the 2-D strain points

4.2 Complete Component Model

The final model that represents the internal strain of the component was a composed of two separate VTK files. One file that contained the 2-D strain data points that were filtered with the 'Delaunay 2D' filter and a second that contained the CAD model representation of the original component. Figure 5 displays the final component with the effect of the 'Clip' filter revealing the internal strain data observed below the weld.

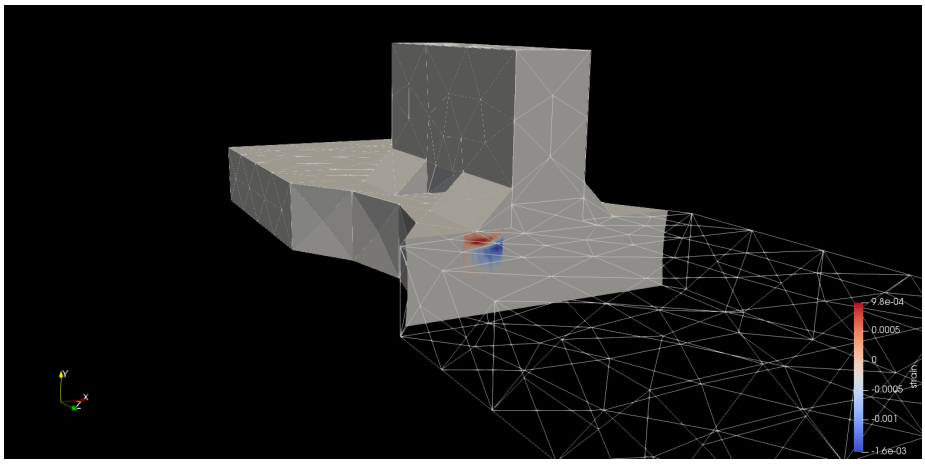

Figure 5: Complete 3-D Representation of component and local 2-D residual strain.

5 Conclusions

The goal of this research project was to visualize residual strain of metallic components with the use of ParaView. This goal was successfully met: the strain data was able to be written into VTK files, uploaded, and analyzed in ParaView with the use of python code, functions, and external libraries. VTK files were found to be the most beneficial and efficient files for uploading the strain data and models into ParaView. Python was the preferable code for working with ParaView and writing the files since ParaView is a Python based platform.

Building on this foundation, efficiencies could be gained through improvements to the code used to write the VTK files. The final component displayed in Figure 5 was composed of two individual VTK files; one file would be ideal for future representation of analyzed components. Moving forward, VTK files and the code that is used to write them should be a prioritized launching point. These files are capable of storing immense data and they are the most compatible file with the ParaView platform do to being out-sourced by the same company Kitware. Depending on the extent of the number of EDD measurements taken of a component, the coordinate location of those measurements could be assigned as points in an 'unstructured grid' VTK file of the part. Then, scalar attribute data values (strain) could simply be assigned to each point that would then be interpolated in ParaView.

Acknowledgements

I would like to thank CHEXS funded by NSF (Grant number DMR-1829070) and MSN-C funded (FA8650-19-2-522) Thank you to Fort Lewis College and Cornell University. I would also like to acknowledge and thank Chris Budrow for used python functions. Thank you to all the other mentors apart of the MSN-C that were not mentioned at the beginning of this report.

References

1. Inc., K. *VTK File Formats* Last accessed 2 August 2019. 2010. `https://vtk.org/wp-content/uploads/2015/04/file-formats.pdf`.
2. Herrera, P. *pyevtk 1.1.1* Last accessed 2 August 2019. 2019. `https://pypi.org/project/pyevtk/#description`.
3. Moreland, K. *The ParaView Tutorial* tech. rep. Version 5.6. Technical Report SAND 2018-11803 TR, Sandia National Laboratories (2018).