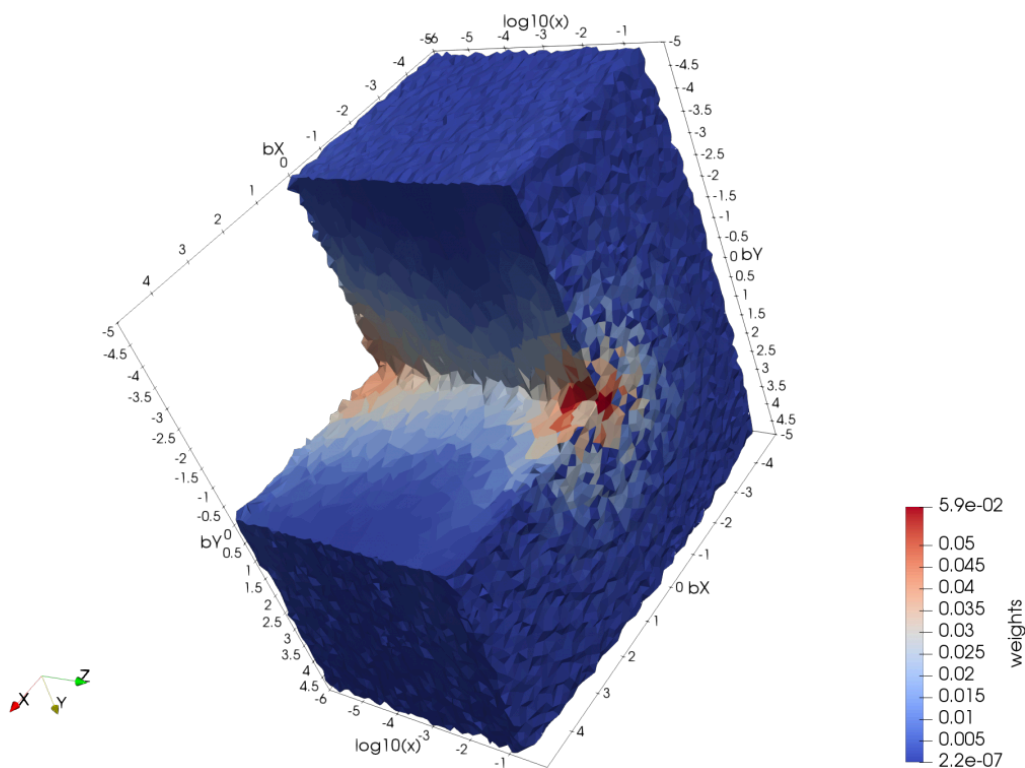


Next-Generation Imaging Filters and Mesh-Based Data Representation for Phase-Space Calculations in Nuclear Femtography

Final Progress Report
SURA Grant No. CNF19-04



Nikos Chrisochodios (PI), Christos Tsolakis, Angelos Angelopoulos, Polykarpos Thomadakis
Computer Science Department, CRTCLab, Old Dominion University
Gagic Gavalian (Co-I) and Christian Weiss (Co-I)
Jefferson Lab

Executive Summary: This is a progress report for SURA CNF19-04 on "Next-Generation Imaging Filters and Mesh-Based Data Representation for Phase-Space Calculations in Nuclear Femtography".

We put together an interdisciplinary project to leverage advanced computational methods from medical imaging for processing phase space distributions in nuclear femtography experiments, with the aim to enable next-generation process simulations, data analyses, and physics model comparisons. The short objectives were: (O1) Implement an n-dimensional Exact Signed Euclidean Distance Transform for image-to-mesh conversion of phase space data; (O2) Implement an n-dimensional mesh-based representation of phase space data using tessellation methods and test it in low-dimensional scenarios ($n=3$); and a longer term objective was (O3) Explore potential physics model comparisons enabled by the new technologies.

Objectives O1 and O2 are completed and implemented within the Parallel Optimistic Delaunay (PODM) mesh software, for efficiency and ease-of-use. The end software product, CNF Image-to-Mesh (I2M) conversion Tool (in shot CNF I2M Tool) delivered to Jefferson Lab. The longer term objective O3 is in progress. However, preliminary data on a spatial distribution of up quarks as a function of proton's momentum fraction carried by those quarks generated by Dr. Sznajder and processed using CNF I2M tool by Dr. Gagik are depicted in Fig. 1. Namely the graph shows b_X and b_Y are the spatial coordinates (in $1/\text{GeV} = 0.197 \text{ fm}$) defined in a plane perpendicular to the nucleon's motion, x is the fraction of proton's momentum and colour denotes probability density for finding a quark at given (b_X, b_Y, x) .

In addition to those stated objectives, this project was very successful with outreach activities. Organized two mini-workshops on Imaging and High Performance Computing to learn lessons and leverage state-of-the-art technologies from Medical Image Computing and Exascale-era and Edge/Cloud-computing communities. The workshops brought together experts (Radiologist, Neurosurgeons, Computational Radiologist and Computer Scientists) from Harvard Medical School and Computational Scientists and Software Systems Engineers from Argonne National Labs (ANL) and VA Tech. The ANL collaboration will help to anticipate future DoE-wide collaborations to meet the exascale computational hardware and low-level software needs of CNF projects. Also, we were able to attract Venture Capitalists (former VA Secretary of Commerce and former ODU's President) with understanding of the landscape for Virginia's transfer of technology and deep understanding of VA economy, respectively. The goal of their participation and continued engagement with this project is to assist JLAB/CNF in building a strong case regarding the potential benefits of CNF related research activities and end-products to create highly skilled jobs in Virginia.

Finally, this project so far involved (and we are in the process of training) six undergraduate and graduate students at ODU —two students are underrepresented in STEM disciplines.

Introduction

The short-term proposed deliverables were: (a) Post-processing of the output of the PARTONS GPD physics model for generation of NRDD/DICOM images and their segmentation; (b) generation of n-dimensional sequence of tessellations for numerical integration; (c) use of n-dimensional tessellations for efficient sampling of physics model and generation of event distributions for physics analysis (d) experiment with 3D Slicer and Paraview open source imaging and visualization systems to display visual representations of the entire space and/or m-slices, with $m < n$; (e) topical workshop including experts in imaging, exascale computing, uncertainty quantification, and nuclear physics to guide further developments

The short-term accomplishments are: (a) JLAB completed the conversion of Nuclear Physics (NP) data, including Nuclear Structure function distributions (see Fig.1), cross sections (see Fig. 2), reaction amplitudes to NRDD/DICOM images which CRTCLab at ODU uses for tessellation of the data, since early May of 2019; (b) CRTCLab completed and delivered to JLAB the Image-to-Mesh (I2M) conversion software for isotropic tessellation of NP data into simplexes¹, this software is portable using Docker technologies and is running from regular laptops and high-end cc-NUMA multicore workstations over three different operating systems: MacOS, Windows and Linux -- in the rest of this document this software is referred in as CNF I2M conversion; CNF I2M conversion software is capable to generate 2D and 3D adaptive isotropic grids or meshes, this capability is completed and actual meshes are delivered to JLAB,

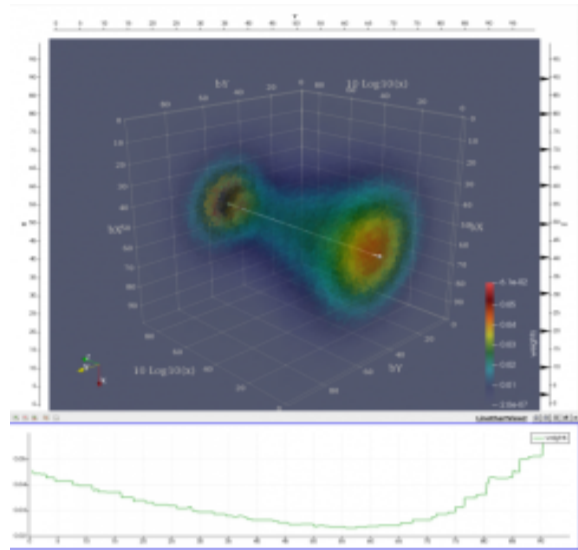


Figure 1: spatial distribution of up quarks as a function of proton's momentum fraction carried by those quarks.

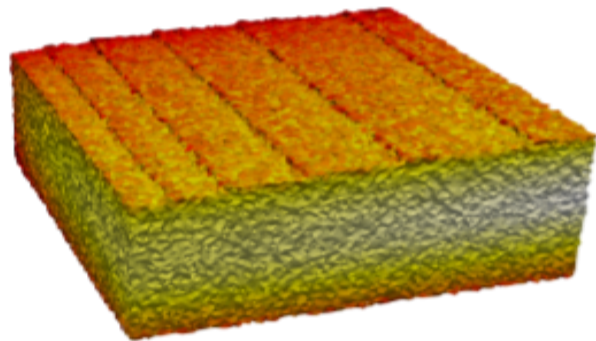


Figure 2: Cross section of Deeply Virtual Compton Scattering as a function of x , t and ϕ

¹ Simplex is an n-dimensional element defined with the minimum number (i.e. $n+1$) points for example in 1D a line segment (2 points), in 2D a triangle, in 3D a tetrahedron, in 4D a pentatope (5 points), etc..

an update of the CNF I2M conversion software with this capability is part of the supplement and will be delivered to JLAB for 2D early November and for 3D early December; (c) Generate one test case of phase based distributions using a known theoretical model from PARTONS and run adaptive tessellation and demonstrate Monte Carlo simulation accuracy and speed; specifically it is computed by Dr. Gavalian that those tessellation are capable to speed up or accelerate analysis by far (see Dr. Gavalian presentation at SURAs CNF Washington DC meeting): “Generating grids using PARTONS took 4 days (1M computations, per grid point), [However] Event generation (10M events) takes ~20 minutes” (d) CRTC and JLAB evaluated two state-of-the-art imaging tools: 3D Slicer and Paraview, for processing and visualization of images and for time being a decision is made to use Paraview; it is very encouraging to see that other SURA CNF projects, for example VATEchs (PI: Nicholas Polys) use Paraview to visualize 4D data which will be very helpful for our followup project --an effort to collaborate and integrate with VATEchs visualization project is initiated from our side; (e) finally organized two mini-workshops: one on Imaging at JLAB (August 2019) and the second at NIA (October 2019) on High Performance Computing.

The long-term goals envisioned (2 years+) are: (a) Development of an HPC imaging infrastructure for efficient simulation and analysis of nuclear femtography experiments, allowing users to implement physics models, generate phase space distributions, constrain model parameters with forthcoming experimental data, and share/communicate results; (b) explore the potential of further interdisciplinary collaboration to address CNF objectives involving new computational methods and longer-term projects.

We are very pleased that we can report a very good progress towards the long-term goals: (a) the CNF I2M conversion software is tested on high end cc-NUMA nodes successfully and preliminary data from a related NSF project indicate that it scales well for more than 16,000 cores, in fact it is capable to generate more than 100 billion tetrahedra within 1000 seconds on about 16300 cores -- the test is performed on ANL HPC machines. Of course, more work is required in order to improve scalability and refactor the code for portability and maintainability; such efforts will be undertaken in the future and will be scheduled based on the overall directions of the CNF efforts in VA. (b) we were very successful to attract a highly-competitive (that involves other DoE labs) network for interdisciplinary collaborations with the two mini-workshops. This capability will allow us to quickly react to deadlines and projects with collaborations in areas that complement our expertise in medical imaging (with Harvard Medical School) and edge-, cloud- and exascale-computing as well as Machine Learning (with Argonne National Labs and VATEch). In addition, we identified specific follow-up action items in anticipation of future and potential CNF needs, for example: (i) data standards --we can leverage technology and lessons learned from HMSs recently supported from NIH project on Imaging Data Commons or (ii) low-level support for power-aware, out-of-core processing of large data sets on exascale-era machines --we can leverage Argo OS developed at ANL by Dr. Pete Beckman -he participated in HPC mini-workshop and there are a number of followup steps in this direction.

In the rest of this document we will briefly provide some background on the CNF I2M conversion tool, the method we used, the portability aspects of the software and future directions.

Background for I2M Conversion: The sequential volume mesh generation methods can be divided into two categories: PLC-based and Isosurface-based. The PLC-based methods assume that the surface ∂O of the volume O (about to be meshed) is given as a Piecewise LinearComplex (PLC) which contains linear segments and polygonal facets embedded in 3 dimensions. The limitation of this method is that the success of meshing depends on the quality of the given PLC: if the PLC forms very small angles, then termination might be compromised. On the other hand, the Isosurface-based methods assume that O is known through a function $F: \mathbb{R}^3 \rightarrow \mathbb{R}$, such that points in different regions-of-interest evaluate F differently. This assumption covers a wide range of inputs such as parametric surfaces/volumes, level-sets and segmented multi-labeled images. Function F can also represent PLCs (albeit without topological guarantees), a fact that makes the Isosurface-based method used in this project a more general approach. Isosurface-based methods ought to recover and mesh both the isosurface ∂O and the volume. Also, since the isosurface is not already meshed into a PLC but it is recovered and meshed during refinement, this method does not suffer from any angle constraints.

The Method for I2M Conversion: The I2M conversion method relies on the well-known Bowyer-Watson kernel [1, 2]. The use of this kernel ensures that the elements will still satisfy the Delaunay property, namely, that the circumball of all elements will not contain any mesh vertex. Specifically, the kernel computes the cavity $C(p)$ of p , which consists of the elements whose circumball contains p . These elements are deleted (because they violate the Delaunay property) and p is connected to the vertices of the boundary of $C(p)$. It can be proved that both the remaining and the new elements satisfy the Delaunay property [1, 2].

The parallel implementation relies on Speculative or Optimistic execution model where insertion of points by different threads needs to be synchronized since cavities intersect at a non-empty set [3]. In this case, only one thread can proceed and the other discards the changes and rollbacks [4,8]. Contention Managers (CMs) decide which threads rollback and which do not. They are inspired by the on-going research on transactional memories [5–7]. Their goal is not only to decrease the number of rollbacks but also to guarantee the absence of livelocks. Two are typically the CMs employed in the mesh generation literature [4, 8]: the Aggressive-CM and the Random-CM. None of these CMs, however, provably guarantees the absence of livelocks. In this project, we use application-specific CMs which (a) provably eliminate livelocks, increasing scalability and guaranteeing correctness on any core count, and (b) throttle concurrency and implement race-to-halt.

Delaunay tessellation is a powerful method for generating meshes, and it has several benefits. Most importantly, it can generalize to N -dimensions, including for $N > 3$. Consequently, it can be

used to generate tetrahedral meshes of 4-dim or greater, data sets. Another benefit is that the n-dimensional simplex mesh generated is memory efficient and can be used to visualize large data sets while keeping the entire mesh in memory.

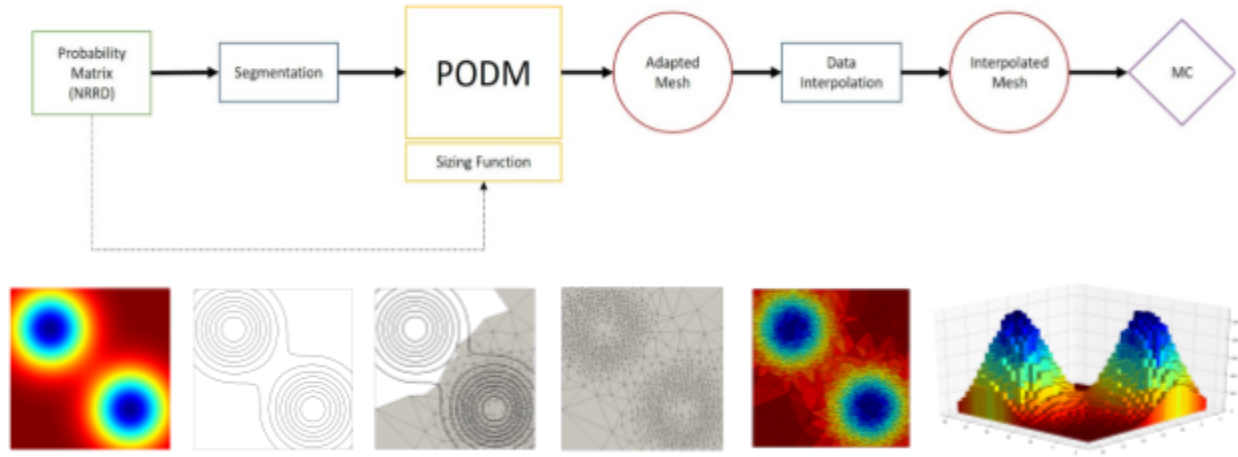


Figure 3: The processing sequence that takes place in order to produce an interpolated mesh from a probability matrix. This sequence is explained in more detail in the following section.

Its parallel implementation relies on low-level locking mechanisms, contention management, and load balancing across multiple processors [9]. I2M conversion allows the user to control the fidelity of the generated mesh, by controlling the size of the generated tetrahedra that compose the mesh. Other advanced options, such as volume grading, which allows for varying tetrahedra sizes in the mesh, are offered. Adaptivity (i.e., mesh size gradation is possible, but the current implementation due to time limitations adopted an ad-hoc approach which will be generalized with a proposed supplement.

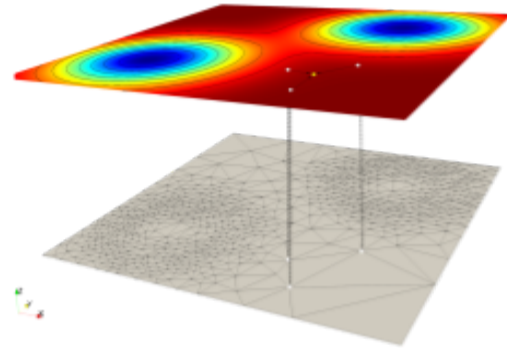


Figure 4 : Evaluating element weights by using cell-averages.

The Software Pipeline for I2M Conversion: The software pipeline is depicted in Fig. 3 and consists of: (A) segmentation filter that gets as an input NRRD image and generates a mask; it requires --due to the nature of NP data-- a much simpler implementation than medical image segmentation filters; (B) I2M conversion tool which takes as an input: (i) a mask and (ii) an NRRD image and generates a graded isotropic mesh based on user-defined metrics for adaptivity, the current 2D and 3D implementations query the NRRD image i.e., use its attributes to define the sizing function; it relies on Parallel Optimistic Delaunay Mesh (PODM) generation method described above; (C) Data Interpolation module, its current implementation uses linear interpolation; its input is: (i) a 2D or 3D mesh and (ii) the numerical values of the input

probability matrix which are transferred to the mesh in order to be used as an input to the subsequent Monte-Carlo simulations see figure 1. There are multiple methods to transfer a solution to a mesh, including interpolation (linear or higher order) to the vertices or elements of a mesh. As a first step towards validating the pipeline, a classic method is used, similar to the one cell-centered CFD solvers evaluate cell averages. In particular, the points of each element are transformed from the mesh (physical) space to the image space. The weight of each point is then evaluated by locating the pixel where each point falls in the probability matrix. These weights are averaged and stored within each element in the output file. Fig. 2 depicts an example. The three white points of a triangle in the mesh are transformed into image coordinates and their weights are averaged. The averaged value is then used as a weight for the original element.

CNF I2M Conversion Software Usage and Workflow: A condensed Docker image containing the modules described above has been created for easy distribution to users, Fig. 5 below depicts the Docker pipeline. The software operates directly from within a Docker container and all processing happens within it. Data is transferred from a user machine to the Docker container via the use of bind mounts or volumes. Input is read from a user's computer, processed inside the Docker container, and then the resulting mesh is written back to the user's computer.

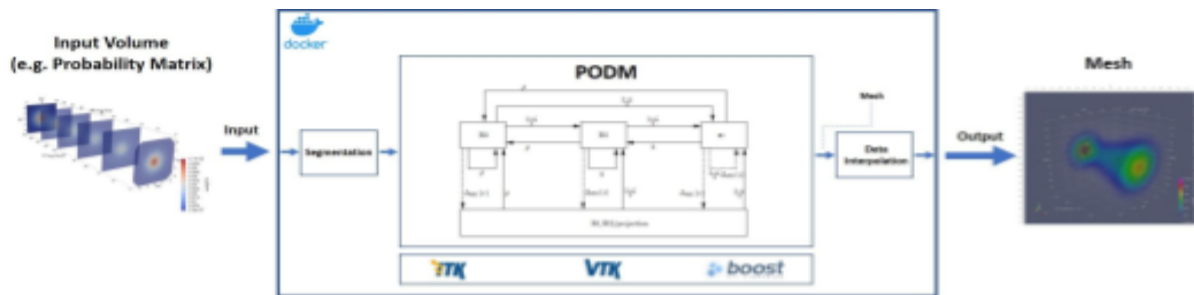


Figure 5: The compartmentalization of the software in a Docker container. The figure on the left represents the input of the software (e.g. a probability matrix). The Docker container in the middle contains the software along with its dependencies pre-configured and installed. While the figure on the right depicts the resulting mesh as the output of the software.

There are many benefits to this approach, including portability, ease-of-use, and isolation (i.e. the isolation offered by Docker technologies, meaning no files are created on the base system). In addition, the usage of Docker ensures the creation of a well-defined runtime environment in which the software has been tested and verified by us. Users can simply load the Docker image, spawn a Docker container, and start using the software immediately. The Docker image contains a lightweight distribution of Debian Linux, PODM, an image conversion tool, and all the dependencies required for the software to run (e.g. ITK and VTK). The Docker image can be used on all operating systems that Docker supports, including Microsoft Windows, Linux, and macOS. Consequently, this software suite does not need to run on a supercomputing cluster. It

also runs from laptops and personal computers with a handful of cores to high-end workstations with several tens of cores.

In order to use the CNF I2M conversion software suite, the user needs to load the Docker image we provide. Then, one needs to place the data they want to have meshed in a folder. This folder will be shared with the spawned Docker container and its contents will be accessible from it. Afterward, the user simply needs to execute a specific Docker “run” command (listed in the user guide linked in the Appendix) which will spawn a Docker container, process the user-specified data, write the output in the data folder, and terminate.

Input/Output: I2M conversion accepts as input a volume of points in N-dimensions (for this project in NRRD format) and a series of command-line parameters. The most important of these parameters is the size of the tetrahedra composing the generated mesh. The size is responsible for both the fidelity of the mesh and its size in memory. Some images are better meshed using small tetrahedra whereas others can be meshed with bigger ones. There are several command-line parameters and are enumerated in detail in the [user guide \(https://crtc.cs.odu.edu/CNF_Tools\)](https://crtc.cs.odu.edu/CNF_Tools) which appears in the Appendix. The output from PODM is an N-dimensional tetrahedral mesh representation of the input volume. Generated meshes are in VTK format and can be easily visualized and post-processed in ParaView.

Future Work: The short term (i.e., within the next 12 months) future directions of this project include:

- **Incorporating Monte Carlo event generation for deeply virtual compton scattering software into JLab HALL-B physics analysis workflow,** some other analysis such as deep inelastic scattering would benefit from this technology.
- **4D Delaunay-based Image-to-Mesh Conversion method for CNF’s NRRD images.** Assuming a closed hyper-surface the method will guarantee faithful geometric and topological approximation. The deliverable will be a portable implementation on a single multi-core Shared Memory High-End Workstations or Nodes which are the building blocks of current and next generation supercomputers. In addition a unified interface (see below) for all 2D, 3D, and 4D dimensions will be implemented. This approach will make the software easier to maintain and improve its performance as needed.
- **Unified *PODM* Interface:** The CRTC has developed the *Tessellate 2D* mesh generator and several variants of the *PODM* mesh generator for different input dimensions. For example, to process 3D input volumes, usage of *PODM 3D* is required. For 4D input volumes, *PODM 4D* is required, and so on. Currently, all of these mesh generators are

standalone, and end users need to receive multiple different executables to use them. This is not ideal and may be cumbersome for the user.

Since the mesh generators share a lot of code and have very similar command-line interfaces, we can create a new application that integrates an arbitrary number of *CRTC* mesh generators under a single executable and command-line interface. This is better from a user experience perspective and it also helps to improve the maintainability of the code. It improves the user experience because the user will only have to use one application and they will be able to choose with a command-line argument which mesh generator they want to use depending on the dimensions of their input volumes. It also improves code maintainability because we will only have to maintain one command-line interface, one system for I/O, one system for pre-processing, and one system for post-processing, instead of multiple ones for each mesh generator. Moreover, this system will be deployable inside a Docker container, offering portability and further improving the end user experience. Therefore, a unified *PODM* interface can be very beneficial for future work.

The architecture of this new integrated system will consist of a common executable that provides an interface for various mesh generators. This common executable will handle command-line argument parsing, I/O, preprocessing of input volumes, and postprocessing of output meshes. There will be a variety of *filters* included, some for pre-processing and some for post-processing. The user will be able to select which ones they want. Current filters include image segmentation and linear interpolation, and more can be added in the future as needed. Part of this effort includes adapting the mesh generators to work as libraries because all of the mesh generators and their dependencies are compiled as shared objects which are linked with the common interface executable. Figure 6 shows the architectural design of this system. This new application is in development, and will integrate the Tesselate2D and PODM mesh generators under a singular command-line interface, with which the user can process n-dimensional input using the mesh generator matching the dimensions of their input.

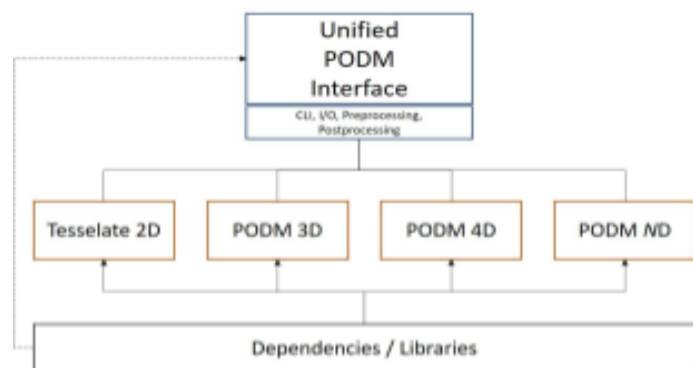


Figure 6: Software architecture for PODM's API

- **Ease-of-use:** There are many ways in which the software suite we have created for this project can be expanded to improve the user experience:
 - a. Incorporate Jupyter notebook support, which would make it easy to create meshes from data and display them instantly in a Jupyter notebook using an easy-to-use interface. This entire extension would be implemented with Docker and will be portable.
 - b. Implement ParaView plugins that would make the software suite immediately and easily usable from within ParaView, which is used to view and post-process generated meshes. This would involve implementing a base plugin for ParaView utilizing a server-client model to establish a communication pipeline with I2M conversion. Implementing this plugin would allow further extension in the form of additional plugins built on top of the base plugin. An example of such a plugin would be one for mesh adaptivity. All in all, these plugins would enable a smooth workflow where users will be able to create meshes and view them without ever exiting ParaView.

- **Machine Learning for tracking particle trajectories in CNF.** Tracking the trajectory of particles in a drift chamber is a time consuming and error prone process. Especially for high luminosity experiments with multi-track final states, the traditional tracking algorithms are computationally intensive, since they need to consider all possible combinations of segments where a collision has been detected (see Fig. 7) . By using machine learning, it is possible to reduce the search space for those algorithms and, as a result, the time and power spent in this process, by filtering out erroneous detections before using the traditional algorithms. This problem can be treated as an imaging problem by representing the trajectories that are detected as lines in 36x112-pixel images. By doing that, we can leverage work done in medical imaging and experiment with Convolutional Neural Networks (CNNs) to detect the valid tracks and reduce the amount of invalid ones. In parallel to this approach we are also investigating a variety of shallow and deep learning models that work on raw data with the aim to narrow our focus down to suitable models, for example ExtraTrees and Dense Neural Networks (DNNs). In addition to the image-like representation, we are envisioning two more for the non-CNN models: In the first one, there is only a single feature per layer in the drift chamber showing which of its wires has made a detection (36 features/sample). In the second, each combination of trajectories detected by two adjacent segments in the



Figure 7 : Sensor detections in drift chambers

drift chamber is represented by a single line, described as a pair of its center and its slope (6 features/sample). Our preliminary results show an accuracy of approximately 96%.

References

- [1] Adrian Bowyer. Computing Dirichlet tessellations. *Computer Journal*, 24:162–166, 1981.
- [2] David F. Watson. Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes. *Computer Journal*, 24:167–172, 1981.
- [3] Andrey Chernikov and Nikos Chrisochoides. Practical and efficient point insertion scheduling method for parallel guaranteed quality Delaunay refinement. In *ACM International Conference on Supercomputing*, pages 48–57, Saint-Malo, France, June 2004.
- [4] Christos Antonopoulos, Xiaoning Ding, Andrey Chernikov, Filip Blagojevic, Dimitrios Nikolopoulos, and Nikos Chrisochoides. Multigrain parallel Delaunay mesh generation: challenges and opportunities for multithreaded architectures. In *ACM International Conference on Supercomputing*, pages 367–376, Cambridge, MA, June 2005.
- [5] William N. Scherer, III and Michael L. Scott. Advanced contention management for dynamic software transactional memory. In *Proceedings of the 24th annual ACM symposium on Principles of distributed computing, PODC '05*, pages 240–248. ACM, 2005.
- [6] Maurice Herlihy, Victor Luchangco, and Mark Moir. Obstruction-free synchronization: Double-ended queues as an example. In *Proceedings of the 23rd International Conference on Distributed Computing Systems, ICDCS '03*, 2003.
- [7] Maurice Herlihy and J. Eliot B. Moss. Transactional memory: architectural support for lock-free data structures. *SIGARCH Comput. Archit. News*, 21(2):289–300, May 1993.
- [8] Démian Nave, Nikos Chrisochoides, and L. Paul Chew. Guaranteed-quality parallel Delaunay refinement for restricted polyhedral domains. *Computational Geometry: Theory and Applications*, 28:191–215, 2004.
- [9] Panagiotis Foteinos and Nikos Chrisochoides. High Quality Real-Time Image-to-Mesh Conversion for Finite Element Simulations. *Journal on Parallel and Distributed Computing*, 74(2):2123-2140, 2014.

Appendix: User Guide

Requirements: OS: Linux, Windows 10 Pro/Enterprise, MacOS Sierra 10.12+ and Docker

Installing Docker

Official documentation:

- [Microsoft Windows](#)
- [MacOS](#)
- Linux
 - [Ubuntu](#)
 - [Debian](#)
 - [CentOS](#)
 - [Other](#)

Note for Running on Windows: Docker on Windows uses Hyper-V VMs to run Linux containers. By default, the spawned VMs use 2vCPUs and 2 GB RAM. If performance is a concern, it is recommended to edit the Docker settings via the GUI to increase the resource allocation for the VMs in order to allow the tessellation tool (PODM) to use more threads.

Getting the Software

The Docker image is located [here](#) (restricted access).

Docker Container Instructions

1. Load the Docker Image.

First of all, the Docker image needs to be loaded. The following command must be used:
`docker load --input [DOCKER_IMAGE_TAR]`

2. Running Currently the docker image includes two tools tessellate and convert_image To use the tessellation tool use:

- On MacOS/Linux
`docker run -v $(pwd):/data/ cnf_tools tessellate [arguments]`
- On Windows with Powershell (recommended)
`docker run -v ${PWD}:/data/ cnf_tools tessellate [arguments]`
- On Windows with command line (cmd)
`docker run -v %cd%:/data/ cnf_tools tessellate [arguments]`

To use the convert_image tool replace *tessellate* with *convert_image* in the commands above.

Tessellation Parameters

A quick way to view all available parameters and brief descriptions for them is to pass the `--help` flag to *tessellate*.

Below are more detailed descriptions of the tessellate parameters:

`--input [filename] (required)`

Input file (3D image). Can be in nrrd format.

`--delta [real] (required)`

Controls the size of the tetrahedrons. Smaller values will lead to finer detail (and often to a more accurate mesh) using more tetrahedrons but will also lead to a greater mesh size.

`--threads [int] (optional)`

How many threads to use.

`--output [filename] (optional)`

The filename of the output mesh. Defaults to `outputMesh.vtk`.

`--plc [filename] (optional)`

If given, the surface of the produced mesh will be saved into filename in the VTK format.

`--volume-grading (optional)`

Creates a volume graded mesh. In this case the size of the tetrahedra is small (controlled by delta) close to the boundary while bigger inside.

`--memory-limit [int in MB] (optional)`

Constrain the amount of memory that will be used (default : 70% of the free memory).

`--verbose-level [0,1,2] (optional)`

Control the level of verbosity. 0 produces no text output while 2 enables extra information (default 1).

`--segment-image [0,1] (optional)`

Enable automatic image segmentation. Defaults to 1 (true). When dealing with non-segmented data this option is required.

`--background-value [real] (optional)`

Which voxel value to treat as background value. If none is desired, enter a value that does not exist in the dataset. (default: 0).

`--linear-interpolation [0,1] (optional)`

Interpolate the image values on the centroids of the final volume mesh. Defaults to 1 (true). It is strongly recommended not to disable this.

`--all-statistics (optional)`

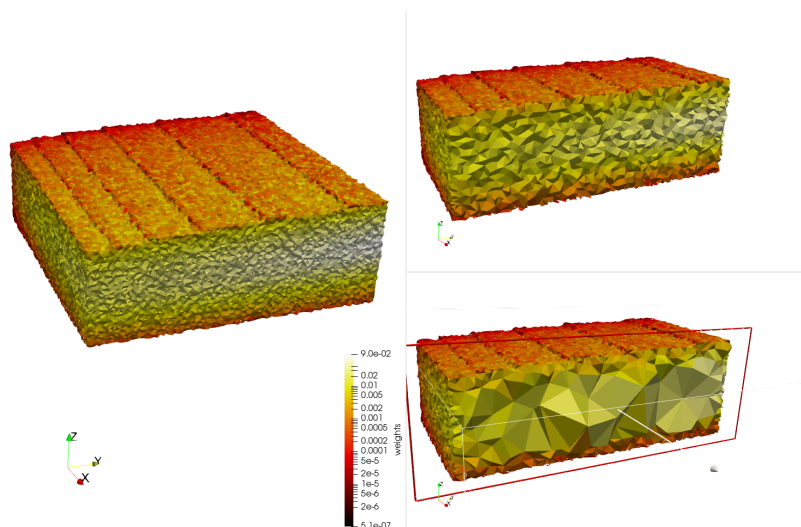
Output all statistics (thread and mesh).

`--thread-statistics (optional)`

Output thread statistics.

`--mesh-statistics (optional)`

Example Dataset: An example dataset can be found [here](#) (restricted access)



Generated using:

Uniform isotropic case (up right)

```
docker run -v $(pwd):/data/ cnf_tools tessellate --input  
./CNF_SHARE/DATA_04252019/OBS_ALU.nrrd --delta 1 --output mesh0.vtk
```

Graded isotropic case (bottom right)

```
docker run -v $(pwd):/data/ cnf_tools tessellate --input  
./CNF_SHARE/DATA_04252019/OBS_ALU.nrrd --delta 1 --output mesh1.vtk  
--volume_grading
```

Appendix: Imaging Mini-Workshop at JLAB

Summary: This was a two-day workshop focused on lessons learned : (i) on Imaging from Surgical Planning Laboratory (SPL) and Image Guided Neurosurgery (IGNS) Lab at BWH, Harvard Medical School; together both labs have more than 40 years of experience and many 100's of



person-years of accumulated research and (ii) from empirical evidence, past experience on economic strengths and weaknesses of various approaches to generating inventions and innovations and

intellectual property and protection --a key part of the process for job creation, with focus on VA ecosystem, state-based incentives and programs for innovation.

The panelist included SPL's founding director, Dr. Ron Kikinis and top executive, Dr. Tina Kapur along with senior researchers like Dr. Steve Peiper and Dr. Andrey Fedorov and consumers of their imaging work like Dr. Alex Golby, Haley Distinguished Chair in the Neurosciences and Professor of Neurosurgery & Radiology at Harvard Medical School. In addition for day two the panel included: (a) Mr. James Cheng (MBA '87) is former Virginia's secretary of commerce and trade; he managed 13 agencies, including economic development, export, tourism, small business, housing, mining and energy. Currently an angel investor and entrepreneur with more than 20 years of experience in information technology and government contracting and (b) Prof. James V. Koch is Board of Visitors Professor of Economics Emeritus and President Emeritus at Old Dominion University.

Follow Up: Start working on ways to define data standards/formats for Nuclear Physics data similar to DICOM in medical imaging. With this initiative we will leverage the work Dr. Fedorov and Dr. Kikinis are starting to develop with a new NIH grant (approximately \$9M) on "cancer imaging to establish and refine best practices for cloud-based imaging data analysis and stewardship". In addition as part of this Imaging workshop Dr. Marek Behr from RWTH Aachen University, Germany visited the PI this summer and met with the PI and JLAB researchers Dr. Gavalian and Dr. Weiss. Like NASA's Computational Materials (CM) 2040 Vision, the common theme is the potential use of 4D mesh conversion and generation techniques for fully unstructured space-time meshes in the context of physics and engineering finite element simulations on modern architectures. For his point of view 4D FEM grid generation leads to "novel computational methods enable not only effortless robust and accurate handling of domain topology changes, but also can exploit both space and time parallelization leading to much

improved scalability”. Current applications of these techniques are tested in Behr's group in the areas of advanced manufacturing - extrusion, molding, casting, additive manufacturing - and mobility - automotive engine components.

Appendix: High Performance Computing Mini-Workshop at NIA

Summary: This was one-day event & panel and is designed to bring together two local research communities: (i) Engineers from NASA/LaRC and (ii) basic research scientists from JLAB along with panelists from DoE's Argonne National Labs (ANL) and VA Universities like VATEch. The goal was to



explore opportunities and identify common HPC issues and common solutions of interest to all sides. A number of such issues were software systems support like ANL Argo OS for exascale-computing and Edge-computing efforts by “moving some of the computations to edge to reduce data output” and VA Tech’s Machine Learning efforts on “Train-less Neural Network Architecture Exploration (TAPAS) tool”. NASA/LaRC Dr. Cara Leckey, Dr. Eric Nielsen and Dr. Ed Glaessgen, addressed HPC, CFD Vision 2030 and CM 2040 Vision efforts, respectively. The common theme is lack of parallel programming tools for highly efficient and challenging codes similar to HPC CNF codes.

Follow Up: As a result of the mini-workshop in HPC, in a first phase CRTC will be producing a publication about the design and implementation abstractions of CRTC’s PREMA runtime system and its mapping to ANL Argobots; with follow-up papers (with Dr. Beckman) on the actual implementation and performance of PREMA on top of ANL’s Argobots threading library and yet a third software and paper (with Dr. Peranau and Beckman) to address implementation aspects and infrastructure to support High Bandwidth Memory (HBM) and out-of-core for PREMA. This feature will greatly improve the performance of applications that utilize huge amount of memory as expected by the exascale-era analysis and processing of images and other challenges related to CFD 2030 and CM 2040. In addition, CRTC and VATEch will customize the TAPAS tool reducing the search space for tracking the trajectory of particles in a drift chamber of current and emerging particle accelerators. In addition, become apparent that the 4-dim grid generation efforts under development for this CNF project could be beneficial to Computational Materials 2040 Vision activities. The PI will further explore this opportunity that can be the beginning of an effort demonstrating the “Usefulness of Useless Knowledge”.