**Blue Waters Petascale Semester Curriculum v1.0**

**Unit 10: Productivity and Visualization**

**Lesson 3: Visualization 1**

**Instructor Guide**

*Developed by* *R. Phillip Bording for the Shodor Education Foundation, Inc.*



*Except where otherwise noted, this work by The Shodor Education Foundation, Inc. is licensed under CC BY-SA 4.0. To view a copy of this license, visit*[*https://creativecommons.org/licenses/by-sa/4.0*](https://creativecommons.org/licenses/by-sa/4.0)

*Browse and search the full curriculum at*[*http://shodor.org/petascale/materials/semester-curriculum*](http://shodor.org/petascale/materials/semester-curriculum)

*We welcome your improvements! You can submit your proposed changes to this material and the rest of the curriculum in our GitHub repository at*[*https://github.com/shodor-education/petascale-semester-curriculum*](https://github.com/shodor-education/petascale-semester-curriculum)

*We want to hear from you! Please let us know your experiences using this material by sending email to* [*petascale@shodor.org*](mailto:petascale@shodor.org)

* Since this is only a 25-minute lesson, the wave equation and PDE ideas could already be covered in previous lessons. Otherwise, start with the image cube of floats as an a prior given data set. The focus of this lesson should only be on movie making implementations and its use cases in different scientific applications
* Instructors should review the materials covered in the presentation slide set and do further readings of the concepts being presented
* This lesson will start with presentation slides that show movies.
* Instructors should use simple random array with lines/objects examples to demonstrate primary color images.
* Understand and present the idea of a post processing step to develop visual graphics animations and know how to use ImageMagik.

**Common Pitfalls for Students and Instructors**

* Watch out for IO, outputs data writes in Fortran is NOT the same as C, Fortran random access is needed for compatibility.
* Beware of making many changes and then testing results. Use an incremental approach – make one change – then test entire program. See what image improvements happened.
* In the beginning, do a paper design of image layout.