Component Specifications

Software Components

An overview of the components in this tool is described below:

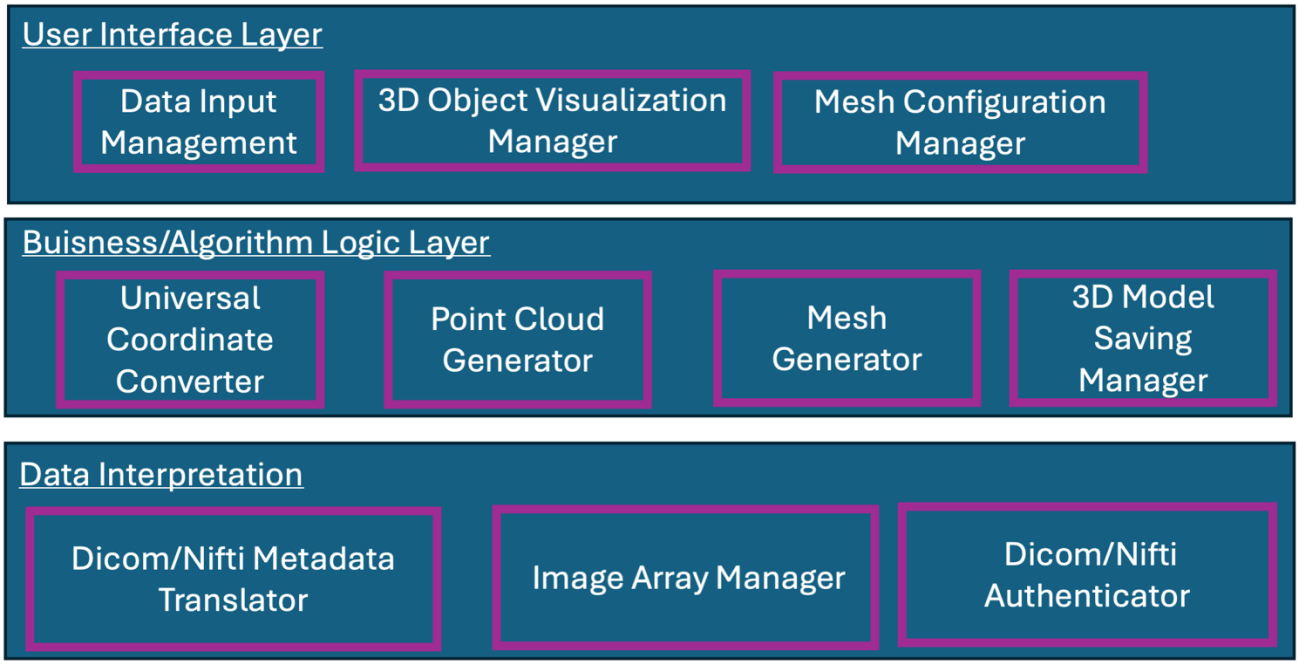

Figure 1. Overview of Components in the Tool. User interface, business/algorithm logic, and data interpretation layers with corresponding software components. ≈

Data Input Management. The data input manager is responsible for importing image data, segmentation, and necessary information on image orientation/views. The image data will be a DICOM file, while the segmentation data can be a nifti, nrrd, or dicom (any file that can store 3D image data). The user will interact with the manager by a file dialog and by typing the necessary information. The input will be the selected path and view names, and the output will be a string of the path for subsequent management, and an internal data structure for processing.

3D Object Visualization Manager. After generation of the mesh, the 3D object visualization manager provides the ability for the user to visualize and interface with the generated 3D object constructed from the imported data. The user will click the desired view, surface or point cloud. The input will be the 3D point cloud data as well as the user selected view option, and the output will be a visual representation of the 3D object. User settings that can be changed include zoom, rotation, and opacity.

Mesh Configuration Manager. The mesh configuration manager allows the user to view the point cloud as a volume mesh. The input will be the 3D point cloud or surface as well as the first pass mesh that was automatically generated. Other inputs are user settings, including mesh density,

resolution, sub-triangulation, smoothing, and geometry optimization. The output will be a refined mesh for export. The user can select export options, which can be the 3D point cloud as .txt file as well as the Delauney triangulation.

Universal Coordinate Converter. The universal coordinate converter involves taking the different views and registering them to a single coordinate system. This coordinate system is dependent upon scanner settings, which are found the metadata of the image input dicom file. The input will be the segmentation data. The output will be a series of matrices containing the (x,y,z) coordinates of each voxel in 3D real space.

Point Cloud Generator. The point cloud generator takes the segmentations and the universal coordinates generated from the universal coordinate converter to generate a 3D point cloud. The inputs are the segmentation masks and corresponding matrices containing the (x,y,z) coordinates of each voxel in 3D space. The output will be a 3D point cloud in an Mx3 matrix, where M is the number of voxels identified as the object of interest in the segmentation mask.

Mesh Generator. The mesh generator takes the point cloud and converts it into a 3D mesh. The algorithm interpolates between points and generates a Delauney triangulation to obtain finite elements. The inputs include the point cloud from the point cloud generator as well as the point cloud density, necessary for interpolation, and the degree of sub-triangulation for increased mesh density/resolution. The output will be an Nx6 matrix containing the (x,y,z) points in the first three columns and connectivities in the last three columns.

3D Model Saving Manager. The 3D model saving manager is responsible for taking the generated mesh and/or point cloud and saving the resulting mesh into a format that is suitable for further analysis and other simulation softwares. The input is the Nx6 matrix with the points and triangulation connectivities from the mesh generator, and the output is the saved file. The file types are selected by the viewer in the mesh configuration manager and includes .stl, .txt, and .nifti.

DICOM/Nifti Metadata Translator. The metadata translator extracts metadata from the headers of the DICOM and/or nifti files to extract the real-space coordinate system, affine transformation, slice thickness, orientation, and voxel size necessary for point cloud generation. The input are the aforementioned files. The outputs are the parsed metadata in a data structure for later use.

Image Array Manager. The image array manager takes both images and segmentation masks that are imported and converts them into numerical arrays (numpy) for subsequent analysis. The inputs are the data structures containing the voxel data, and the output is a standardized array that is a float. The image array manager will also ensure the datatype is consistent (e.g., 8-bit vs. 12-bit vs. 16-bit) as all are used in medical imaging.

DICOM/Nifti Authenticator. This component validates that the paths input by the user correspond to real DICOM/Nifti files that are interpretable by necessary libraries and ensures the they are properly formatted and free of corruption to prevent errors during processing. The inputs

are the uploaded image/segmentation mask files. The outputs are validation status and error messages if errors are detected.

Interactions to Accomplish Use Case

The interactions between these components to accomplish use case 1 and 2 (Dicom and Mask Upload) are described below:

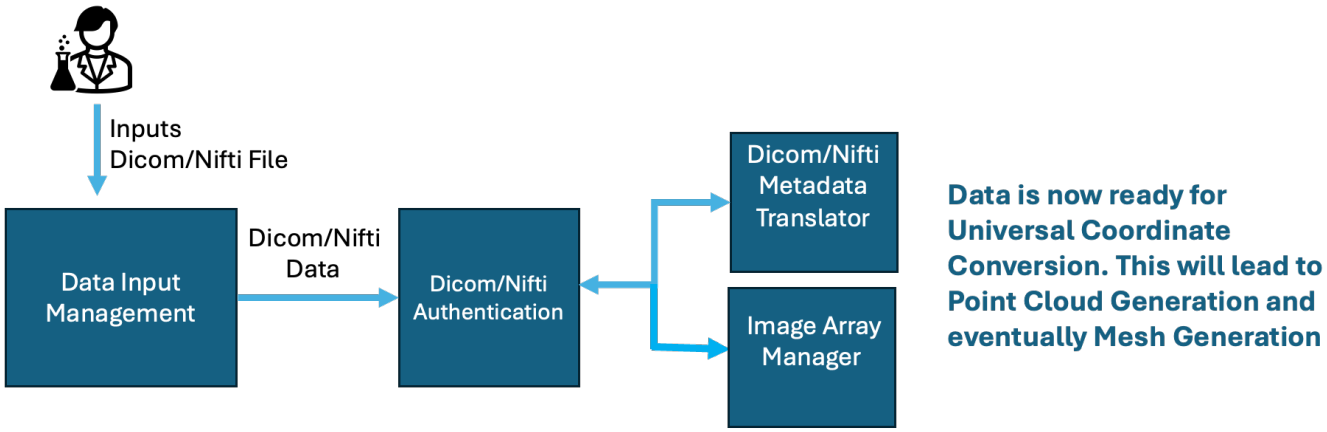

Figure 2. Interactions to Accomplish Use Case 001. The user interacts with the data input manager, which is ultimately fed into the authenticator prior to being input into the metadata and image array translators.

Use Case 001: Image Stack Upload.
Actor. A research engineer or scientists with access to cardiac imaging and segmentation stacks.
Description. This use case involves uploading image stacks from a DICOM format from multiple views to generate a 3D object.

Component Interaction. To accomplish this use case, the user will interact first with the data input manager. This will prompt the user to select the folder containing all the DICOM files for one specific view using an interactive file dialog. The user will repeat this for each view and label it (axial, sagittal, coronal) accordingly. The DICOM/Nifti authenticator will ensure the files are in a DICOM format. This will also ensure that the files are not corrupted and meet required specifications and pass validation status back to the data input manager or an error message. The output will be a confirmation of data upload. After upload the DICOM/Nifti Metadata translator will take the necessary information from the DICOM header, including slice thickness, voxel size, image orientation, and the affine transformation matrix. This will be stored in a data structure for analysis. In parallel, the input array manager will take the necessary voxel data and convert it from the native image data to float in a numerical array. The universal coordinate converter will take the data structure from the metadata translator to generate matrices corresponding to the X,Y,Z position of each voxel. This will be registered with the output of the image array manager. The point cloud generator will take segmentation masks and the output of the universal coordinate converter to generate a point cloud. The location of which X,Y,Z coordinates are derived from the segmentation mask. This is converted to an Mx3 array, where M is the number of points. The mesh

generator interacts with the 3D point cloud generator to derive the mesh using Delauney subtriangulation. The 3D object visualization manager will take the Delaunay triangulation and coordinates for visualization for user interaction. This will interact with all components discussed before (3D point cloud generator, mesh generator) based on user inputs. The saving manager will take the information from the mesh configuration manager, point cloud generator, and mesh generator for saving. The information on which file type comes from the mesh configuration manager.

Preliminary Plan

1. Create dummy data from an open repository including making segmentation
2. Develop Import Manager for Dicom and Nifti Files including a means for authentication.
3. Develop extraction function for Dicom and Nifti Image and Metadata
4. Develop function to use metadata to convert image data to universal coordinates
5. Develop Function to generate point cloud and saving the Euclidian coordinates.
6. Develop mesh algorithm using a combination of open source packages (e.g. MeshPy, Pymesh, ect.)
7. Develop a means for the user to configure one or two of the inputs to the algorithm (e.g. alpha parameter)
8. Develop means to display and save mesh data