Functional Specifications

Background: Mechanical and electrophysiological cardiac simulations are often done in three-dimensions using finite-element analysis and other analytic methods. However, the vast majority of clinical data (e.g., MRI, CT, Ultrasound, etc.) acquire data in 2D. In the specific case of MRI, 3D volumetric imaging is available, but its current use is only for the brain. In all situations, segmentations of relevant anatomical structures are performed in 2D, where each slice represents a plane in the orthogonal direction. It is not trivial to take these 2D image segmentations and generate 3D data. As such, this project aims to create a package and toolkit which allows for generation of a 3D mesh from 2D data. Importantly, this package will accept image stacks and segmentations from different views (e..g, axial, sagittal, coronal, short axis, 2-chamber, 3-chamber, and 4-chamber) and register them into a single 3D object. To our knowledge, this is the first package in the cardiovascular computational regional space to have such a feature and is necessary for the generation of higher quality 3D representations. This may also potentially mitigate aliasing artifacts. Lastly, to make this package usable, a graphical user interface option will be developed for image importation, mesh generation, and output viewing. The proposed pipeline is below:

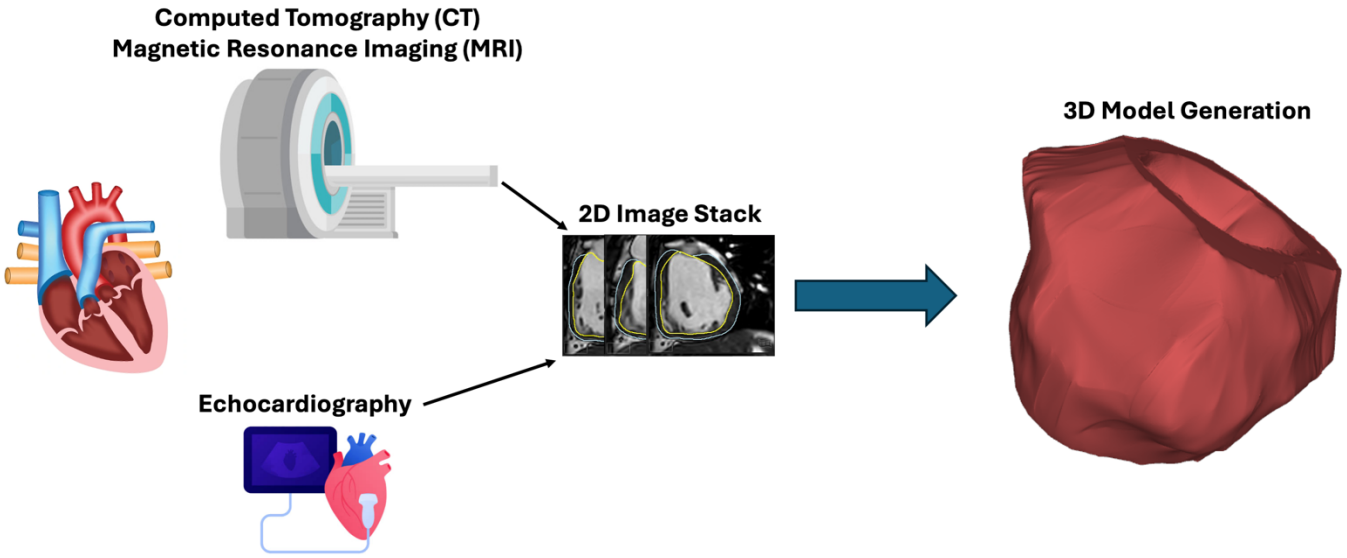

Figure 1. Proposed Software Toolkit. The proposed software toolkit will accept any image stack with its segmentation and automatically generate a 3D mesh for subsequent modeling.

User Profile: This software package will target two user groups. One group is research engineers who specialize in cardiac computational simulations incorporating real clinical data and spans researchers who conduct finite element, computational fluid dynamic, electrophysiological, and other related simulations. This group is projected to have computational flow or electrophysiology simulations. This group is projected to have programming experience, especially with Python. Another group will be researchers who specialize in lower-level cardiac science. These users are projected to not be as experienced with programming and will require either a user interface or clear documentation to run the package without any programming experience.

Use Cases

Use Case ID:	**Use Case 001**
Use Case Name:	**Image Stack Upload**

Actor:	A research engineer or basic scientist, who has access to clinical cardiac imaging and associated segmentation stacks.
Description:	This use case describes uploading imaging stacks in the Dicom format. Importantly, this use case will cover: 1. Uploading imaging stacks from multiple views and 2. Uploading an image stack from one view.
Preconditions:	1. User has "pip" installed the package 2. User has run the package 3. User has an image stack from the same view in the dicom format.
Postconditions:	1. Dicom stacks uploaded to tool 2. Indication of successful upload
Priority:	This is a high priority functionality for this tool as the image data and metadata are required for 3D object reconstruction.
Frequency of Use:	Frequency depends on the amount of patients/image stacks, but each 3D object is going to require at least one upload.
Normal Course of Events:	1. User prepares image stack including make sure the folder only has Dicom images from one view. The number of views/stacks used should equal the amount of folders prepared. 2. User runs the tool. 3. The user will then be prompted to enter which views they will upload. 4. The user enters the corresponding name of each view, e.g., [axial,sagittal,coronal]). 5. A file dialog appears, and the user selects the folder with the axial stack. 6. Another dialog appears, and the user selects the folder for the sagittal stack. 7. Another dialog appears, and the user selects the folder for the coronal stack. 8. The tool prints success for the upload.
Alternative Courses:	Note: Changes to steps written in red Use Case 001 AC 1 1. User prepares image stack including make sure the folder only has Dicom images from one view. The number of views/stacks used should equal the amount of folders prepared. 2. User runs the tool.

<table>
<tr><td></td><td>3. The user will then be prompted to enter which views they will upload.
4. The user enters [axial,sagittal].
5. A file dialog appears, and the user selects the folder with the axial stack.
6. Another dialog appears, and the user selects the folder for the sagittal stack.
7. The tool prints success for the upload.

<u>Use Case 001 AC 2</u>
1. User prepares image stack including make sure the folder only has Dicom images from one view. The number of views/stacks used should equal the number of folders prepared.
2. User runs the tool.
3. The user will then be prompted to enter which views they will upload.
4. The user enters [axial].
5. A file dialog appears, and the user selects the folder with the axial stack.
6. The tool prints success for the upload.</td></tr>
<tr><td>Exceptions:</td><td><u>Use Case 001 EC 1</u>
1. User prepares image stack including make sure the folder only has Dicom images from one view. The number of views/stacks used should equal the amount of folders prepared.
2. User runs the tool.
3. The user will then be prompted to enter which views they will upload.
4. The user enters [axial,sagittal].
5. A file dialog appears, and the user selects a folder for the axial stack that does not have any Dicom (.dcm) files
6. The tool gives the following message: “Error no Dicom files found”
7. The user will not be able to select the sagittal folder.</td></tr>
<tr><td>Special Requirements:</td><td><table>
<tr><th>Req ID</th><th>Short Text</th><th>Long Text</th></tr>
<tr><td>TOO-1</td><td>Input Format</td><td>The tool shall only accept Dicom (.dcm) and nifti image files.</td></tr>
<tr><td>TOO-2</td><td>Input Number</td><td>The tool shall accept a maximum of 3 views.</td></tr>
<tr><td>TOO-3</td><td>View Names</td><td>The tool shall only accept the following names for views:
Sagittal
Axial
Coronal</td></tr>
</table></td></tr>
<tr><td>Assumptions:</td><td>This use case assumes that the tool will be a headless program.</td></tr>
<tr><td>Notes and Issues:</td><td>This use case may change if a GUI is developed.</td></tr>
</table>

Use Case ID:	**Use Case 002**
Use Case Name:	**Segmentation Upload**

Actor:	A research engineer or basic scientist, who has access to clinical cardiac imaging and associated segmentation stacks.
Description:	This use case describes uploading imaging stacks in any 3D image format, including nifti and nrrd. Importantly, this use case will cover: 1. Uploading imaging stacks from multiple views and 2. Uploading an image stack from one view.
Preconditions:	1. User has "pip" installed the package 2. User has run the package 3. User has an image stack from the same view in the nifti or nrrd format.
Postconditions:	1. Dicom stacks uploaded to tool 2. Indication of successful upload
Priority:	This is a high priority functionality for this tool as the image data and metadata are required for 3D object reconstruction.
Frequency of Use:	Frequency depends on the number of segmentations but each 3D object is going to require at least one upload.
Normal Course of Events:	1. User prepares image stack including make sure the folder only has nifti/nrrd segmentations from one view. The number of views/stacks used should equal the number of folders prepared. 2. User runs the tool. 3. The user will then be prompted to enter which views they will upload. 4. The user enters the corresponding name of each view, e.g., [axial,sagittal,coronal]). 5. A file dialog appears, and the user selects the folder with the axial stack. 6. Another dialog appears, and the user selects the folder for the sagittal stack. 7. Another dialog appears, and the user selects the folder for the coronal stack. 8. The tool prints success for the upload.

Use Case ID:	**Use Case 003**
Use Case Name:	**Point Cloud Generation**

Actor:	A research engineer or basic scientist, who has access to cardiac MRI image stacks.
Description:	This use case describes creating a point cloud from the uploaded image stacks
Preconditions:	1. User has "pip" installed the package 2. User has run the package 3. User has uploaded image stacks to the tool
Postconditions:	1. Point Cloud generated and displayed 2. Points saved in a .txt file
Priority:	This is a high priority functionality for this tool as the points from the point cloud are needed
Frequency of Use:	Frequency depends on the amount of patients/image stacks, but each patient will have an associated point cloud (could use data from multiple view)
Normal Course of Events:	1. User will enter in command to create point cloud 2. Tool will prompt the name for the .txt file to save the points 3. Point cloud displayed
Alternative Courses:	N/A
Exceptions:	N/A
Special Requirements:	<table><tr><td>Req ID</td><td>Short Text</td><td>Long Text</td></tr><tr><td>TOO-4</td><td>Save Point Cloud</td><td>The tool shall be able to save point clouds in a .txt format.</td></tr></table>
Assumptions:	This use case assumes that the tool will be a headless program.
Notes and Issues:	This use case may change if a GUI is developed.