Using HPC tools to create 3D tissue models for visualization purposes:

Best Practice Guide

June 2023

EURO CZECHIA

CASTIEL

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Content

- Current workflow (Hospital, IT4Innovations, Misterine)
- Tasks and requirements
    - Specify the requirements to be able to run the current workflow from the command line.
    - Describe the process to allow the cutting of mesh or volume.
    - Describe the process for adding suitable materials for a 3D mesh or volume.
    - Describe the possibilities for reducing geometry topology to accommodate the end devices' needs (AR, VR).
- Conclusion
- References

Current workflow

- A medical doctor from a hospital provides computed tomography (CT) data of the abdomen region or data on different body parts.
- IT4I processes the CT data. The process is as follows:
    - IT4I uses deep learning approaches and provides automatic tissue segmentation from medical images. Such models, e.g., for performing 3D image segmentation, usually combine a large number of labelled datasets with training on multiple GPUs on HPC to provide models of desired quality in a reasonable time.
    - IT4I has created a service that can use the deep learning model to automatically segment desired tissues as an AI-based assisted annotation service using IT4I's infrastructure. This AIAA extension is created in 3D Slicer [1] software. The service collects the data after automatic segmentation and validation by medical doctors and provides HPC-based training for new models or enhances existing models by fine-tuning them.
    - IT4I then uses the 3D Slicer's built-in facilities for post-processing (if necessary) to export 3D models of tissues in standard formats such as OBJ or STL.
    - This 3D model is then imported into Blender [2]. Further operations are performed if required (such as setting origin, creating textures, mesh modification, etc.). The result, which is ready for visualisation, is then exported again as a 3D model with materials, e.g., in FBX format.
- Misterine: Misterine Studio can be used to view the model in augmented reality (AR). The procedure for providing 3D data in AR is as follows:
    - Create a project with a process. Provide a description of the process and tasks, the steps to be followed to view the model in AR, and any warnings (if any).
    - Add a task and convert it to a 3D scene. Open the 3D scene. Import the 3D model. Add a tracking marker to achieve proper positioning of augmentations in the context of the physical environment or add an interactive anchor to let the user roughly position the scene into their physical environment via an interactive process.
    - Deploy the process on a public server and view it in the Misterine app [3] in AR on a mobile device.

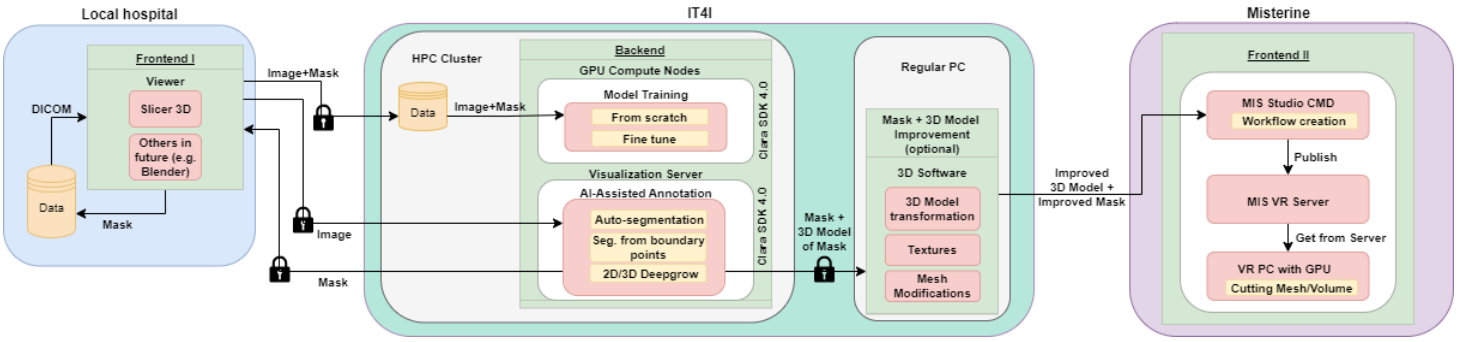


Figure 1 Workflow of medical image processing by IT4I and AR experience in Misterine Studio.

Tasks and requirements

- Running the current workflow from the command line.
    - Converting the functionality of the IT4Innovations AIAA extension workflow to a scripting command-line workflow:
        - Import the necessary modules and packages that are required for functionality.
        - Create a Python or Bash script to take the input and output parameters that the script will need:
            - The input will be the CT scan in Nifti or DICOM format.
            - The parameters that need to be set, for example, can be a user account, password, project ID, ssh key, model name, etc.
            - The output mask data will be generated.
        - Write a script to get resources on the cluster and create an SSH tunnel for the transportation of data over an encrypted SSH connection.
        - Write a script to run the inference on this CT data using the model weights and save the output masks.
        - Test the script.
    - Converting a Misterine Studio UI workflow to a scripting command line workflow:
      Here are some general steps that can be followed:
        - Analyse the UI workflows: Determine the steps required to perform the process and tasks.
        - Write and test the script: We can use scripting languages like Python or Bash to write the script. The script should be written for every type of action and should be able to handle errors.
        - Run the script: Run the script on the command line with appropriate parameters (like file inputs and outputs and values for various variables). It is also possible to automate the running of scripts on a schedule or in response to events.

  The above two workflows can be combined into one script and serve as a single workflow.

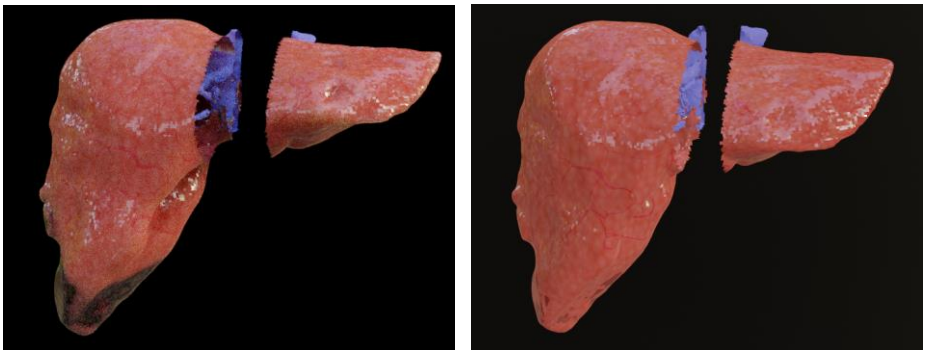
- Add the possibility of slicing the model in any direction using a cutting 3D object (like a plane) and viewing the internal structure with proper colours or textures.
    - Cutting Mesh:

  Below are general steps to cut the model:
        - Select the mesh object you want to cut.
        - Use a cutting tool (like the "Plane Cut", "Slice", "Knife Tool", "Cut Faces", or "Boolean") to draw a cutting plane that intersects with the mesh.
        - Adjust the position and orientation of the cutting plane as needed.
        - Execute the cut operation to create two or more separate mesh objects.

    For example, in Blender, below are a few ways of achieving it:
        - Select the vertices with the lasso tool (or box tool) in edit mode and separate them by selection.
        - Using the Boolean operation with any other 3D cutting mesh (closed surface).

- Add a loop cut on the surface, select vertices, and separate the selection.
- Use the knife tool to add vertices to the model. Select the required faces and separate them.
- Use a 3D cutting mesh, for example, on a plane. Solidify the cutting mesh (if not already) to add a little thickness (like 0.000001 m). Boolean this with the 3D model. Select one of the parts and press the shortcut to select all the linked vertices (or select both the cutting plane and model and go to edit mode and select by side of the active local axis or by normal). Separate the mesh with the separation tool by selection.
- Select the bisect tool and cut the mesh. Select loops by inner region and separate by selection.
- Or cut using a node structure. Separate geometry by selecting vertices in the direction of the normals of the plane.

The mesh should now be cut in half. The vertices, edges, or faces on either side of the cut might need to be adjusted to create a clean seam.



(a) (b)

Figure 2 Cutting Mesh (a) Render with Cycles Engine (b) Render with Eevee Engine in Blender.

Also, for example, in 3D Slicer (open-source software), the mesh can be cut with a plane using the Markups module to create a plane and Dynamic Modeler to cut the model using the plane (uses vtkClipClosedSurface and vtkClipPolyData from vtk). Refer to [4].

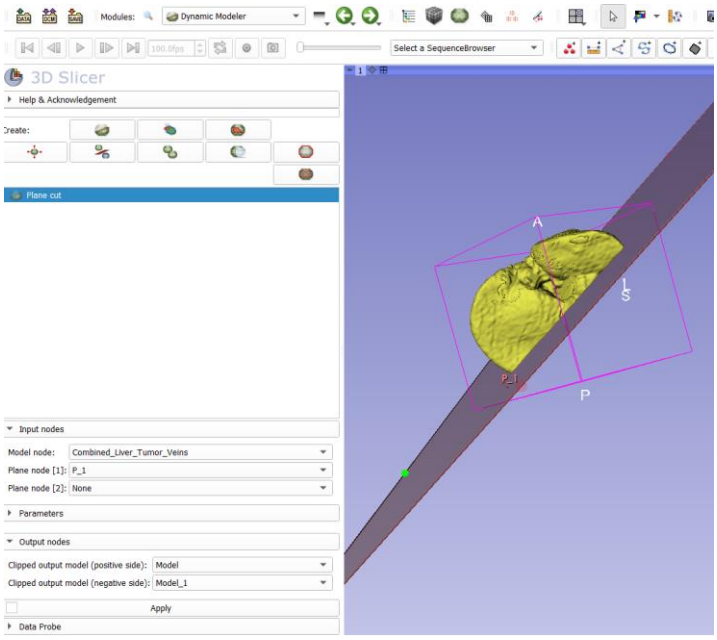


Figure 3 Cutting Mesh with a plane in 3D Slicer.

Clipping (instead of cutting) in 3D Slicer [5]:

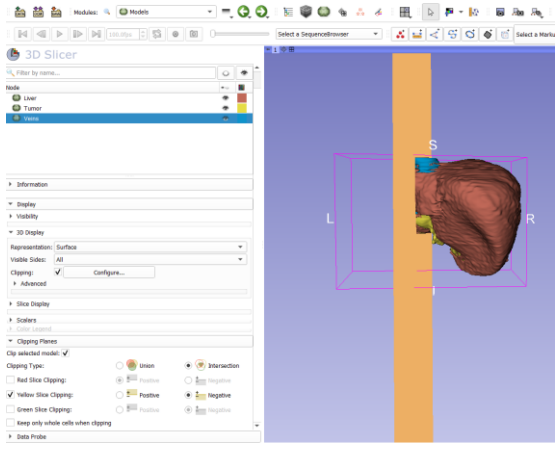


Figure 4 Clipping Mesh with a plane in 3D Slicer.

- Cutting Volume:

  The segmentation mask from Slicer represents a regular volume grid with information about whether the voxel at a specific location is either empty or full and what tissue it represents. This data can be in a format like nifti or nrrd, which can save the 3D voxel data.

  Below is an example of an interactive image cropping widget built using VTK (a library for scientific visualisation). The widget allows users to define a rectangular region of interest within an input image and then crop that region. It cuts in perpendicular directions to the box-shaped region of interest, i.e., it crops the region of interest.

  Use the gl-matrix library to perform vector and matrix calculations.

- Load the 3D dataset into VTK and create a vtkVolume object to visualize it.
- Create a render window with vtkFullScreenRenderWindow.
- Create an instance of the vtkImageCroppingWidget and set its input to the vtkVolume object.
- For volume rendering, use vtkVolumeMapper and create color and opacity transfer functions.
- Move the cropping widget by dragging the handles with the mouse to define the desired cropping region.
- Create vtkPlane objects with normal and origin properties.
  The normal is calculated with information about the volume object's dimensions and the current plane's position. These normal properties are then rotated by the volume data rotation.
  Origin is calculated by multiplying the current plane's coordinates by the volume object's spacing property and adding the volume object's origin property.
- Add the vtkImageCroppingWidget to the VTK render window interactor so that it can be interacted with by the user and visualize the cropped volume.
- Refer to [6].

  Or in Python:
  Load the volume data into a grid (array-like object). Use a library for handling volume data (like pyopenvdb) to manipulate volume data based on the ROI coordinates.

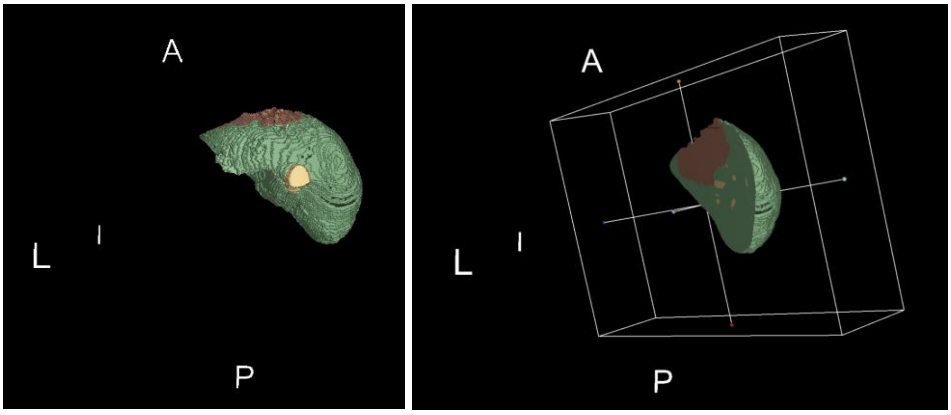


Figure 5 Cropping Volume.

Also, for example, in 3D Slicer, below are a few ways of achieving it.

Cutting volume in 3D Slicer software [7]:

- To simply cut a 3D volume in Slicer, use the Scissors tool and fill inside by changing the editable area.
- To split the 3D volume into two: Create a segment using the Scissors tool. Apply logical operators like Copy, intersect, Subtract, etc. on the segment and 3D volume.

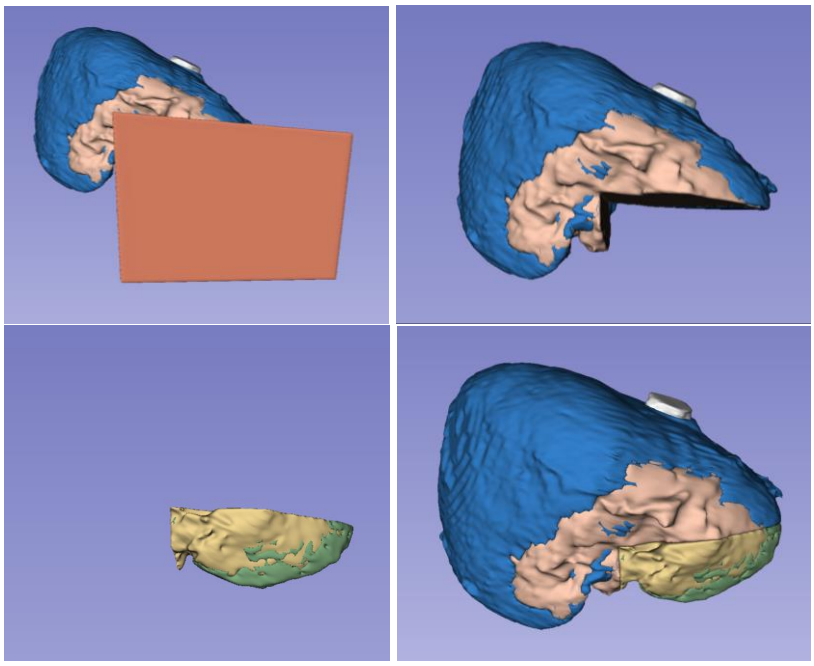


Figure 6 Cutting 3D Volume in Slicer using Scissors and Logical Operators.

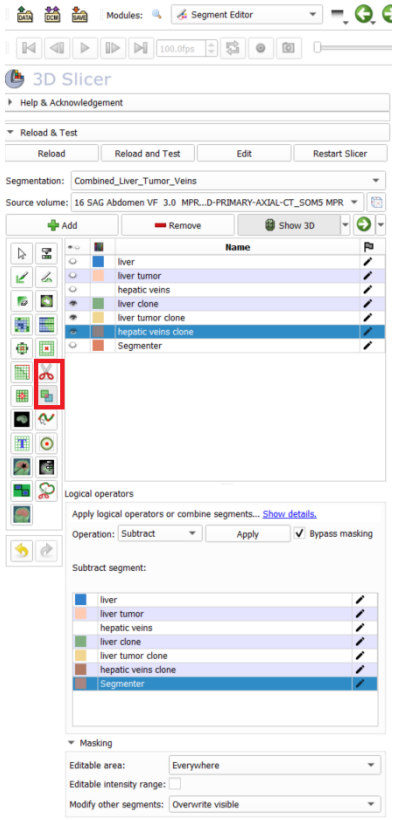


Figure 7 3D Slicer interface for cutting the volume.

Cropping volume in Slicer using Surface Cut and Mask Volume from the SegmentEditorExtraEffects extension. Refer to [8].

```
mask = (maskArray.astype(float) - maskMin) / float(maskMax - maskMin)

inputArray = slicer.util.arrayFromVolume(inputVolumeNode)

if operationMode == "FILL_INSIDE_AND_OUTSIDE":
    # Rescale the smoothed mask
    resultArray = fillValues[0] + (fillValues[1] - fillValues[0]) * mask[:]
else:
    # Compute weighted average between blanked out and input volume
    if operationMode == "FILL_INSIDE":
        mask = 1.0 - mask
    resultArray = inputArray[:] * mask[:] + float(fillValues[0]) * (1.0 - mask[:])

slicer.util.updateVolumeFromArray(outputVolumeNode, resultArray.astype(inputArray.dtype))
```

Figure 8 Script that involves manipulating volumes.

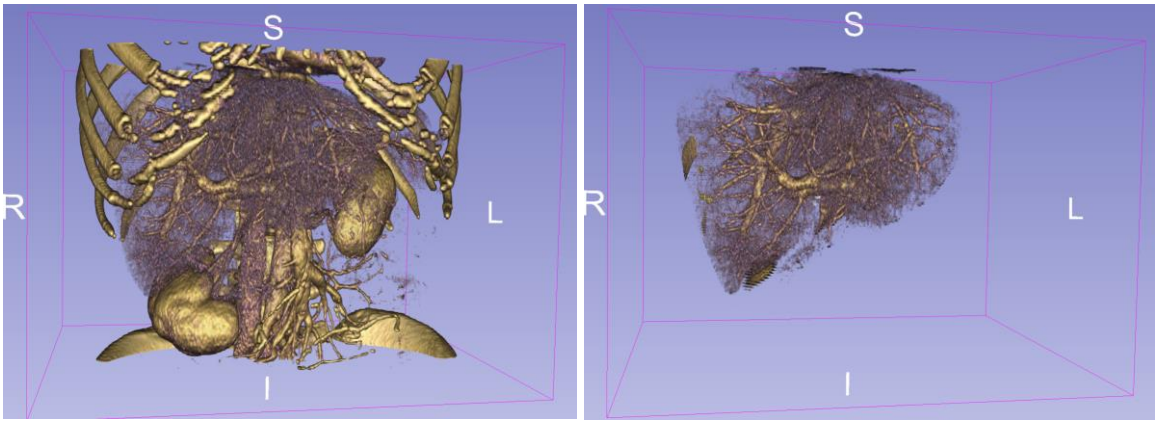


Figure 9 Cropping volume in 3D Slicer using the SegmentEditorExtraEffects extension.

Cropping volume in Slicer using Crop Volume module with rectangular ROI:

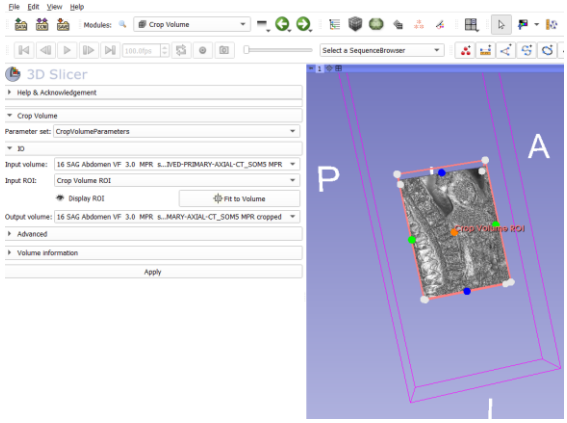


Figure 10 Cropping volume using the Crop Volume module in 3D Slicer.

- Materials

Seamless tileable texture:

- Using medium-quality resolution image textures (2K) or lower to improve workflow performance.
- Use diffuse texture and PBR channels (diffuse, normal, height, roughness, specular, depth, displacement, albedo, etc.) to create a material.

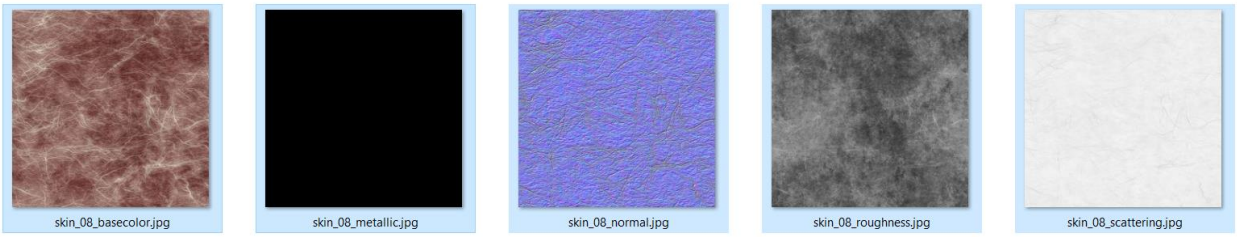


Figure 11 Texture maps of skin flesh.

Procedural material:

Create procedural textures and apply them to the 3D mesh of the organ to make it look more realistic.

Procedural materials can be used for texturing the 3D meshes instead of materials with standard image textures. The benefits of using procedural material are flexibility, efficient memory usage, scalable textures, and the fact that it can be generated at any resolution without losing quality.

- 3D Mesh

  Below is an example of a procedural texture using nodes in Blender:

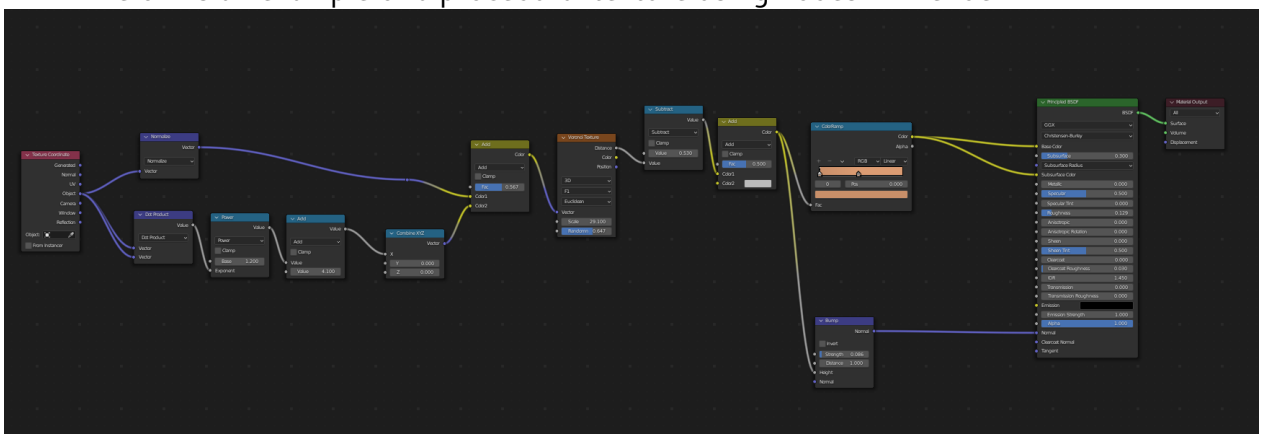


Figure 12 Example of procedural texture material for a 3D mesh.

Figure 13 Examples of different procedural materials.

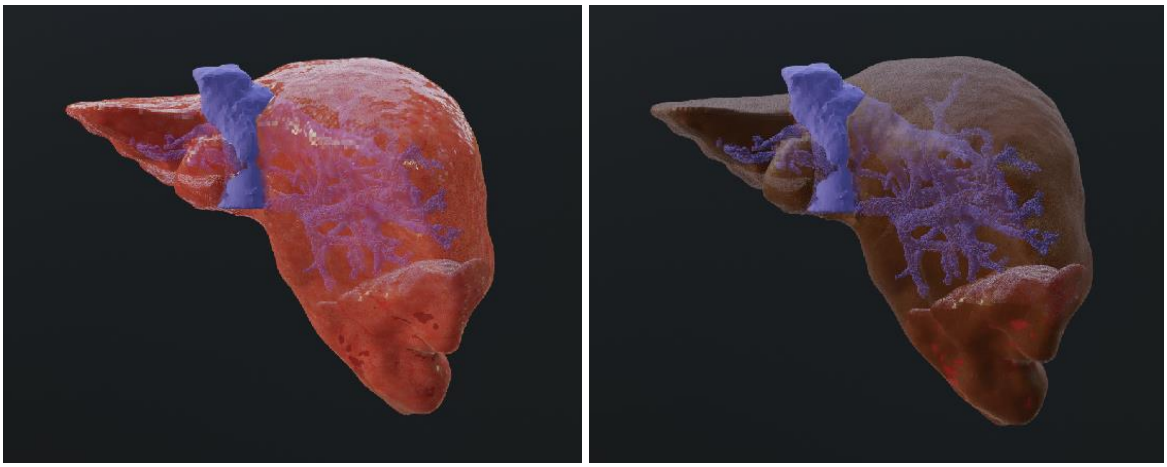


Figure 14 Different procedural materials applied to 3D meshes.

- Volume
  - Apply different colors to different densities.

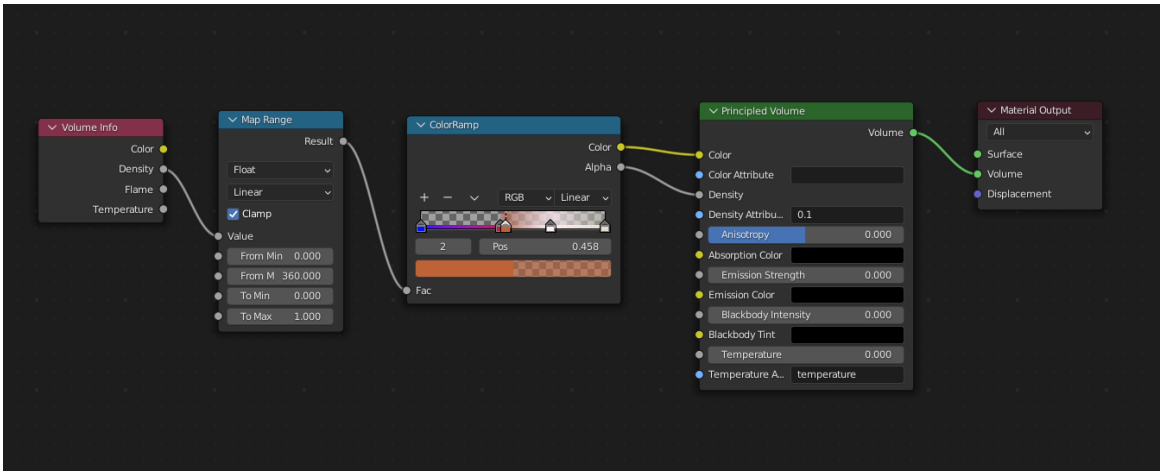


Figure 15 Example of procedural material for volume.

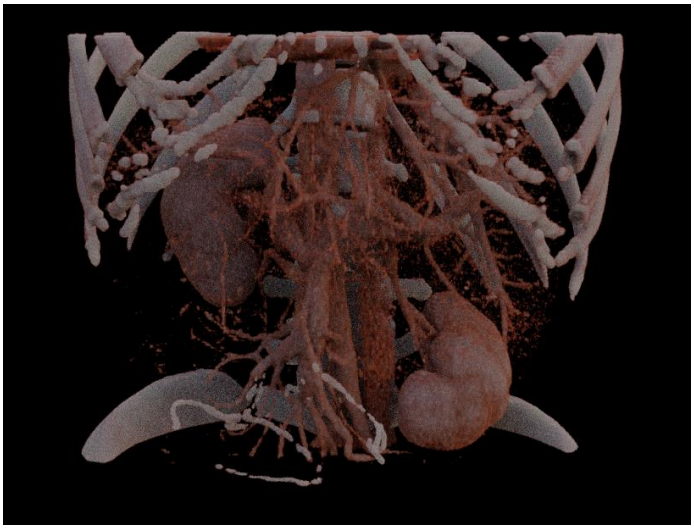


Figure 16 Material Applied to a Volume.

- Limit the mesh triangles that are feasible to be used for AR on mobile devices:

  Limit the number of triangles in the 3D mesh to up to 2 million to improve performance on mobile devices. Use the decimate option within Misterine Studio or other software (like Blender) to limit the number of triangles.

  Below are different ways of limiting the vertices:

  - Using the decimate procedure to merge vertices progressively, taking the shape of the mesh into account:

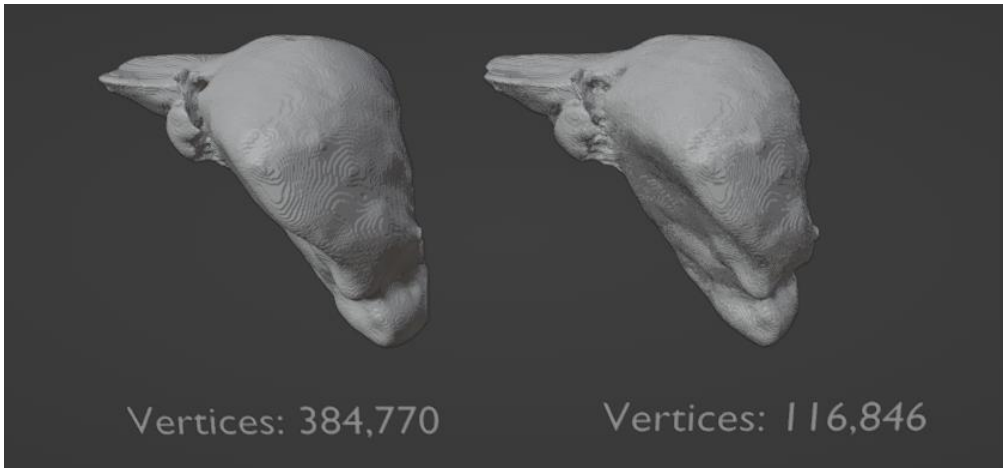


Figure 17 Decimating the 3D Mesh.

  - Create a pipeline in Blender with geometry nodes using the Signed Distance Function (SDF) to remesh the 3D model and control the mesh triangle count and model topology. The mesh is converted to SDF, then it is possible to do mathematical operations on this SDF, and then the SDF field is converted back to mesh. The resolution can be changed to control the vertices.

    The final mesh is also smoothed in this pipeline.

Figure 18 Node structure to change topology using SDF.

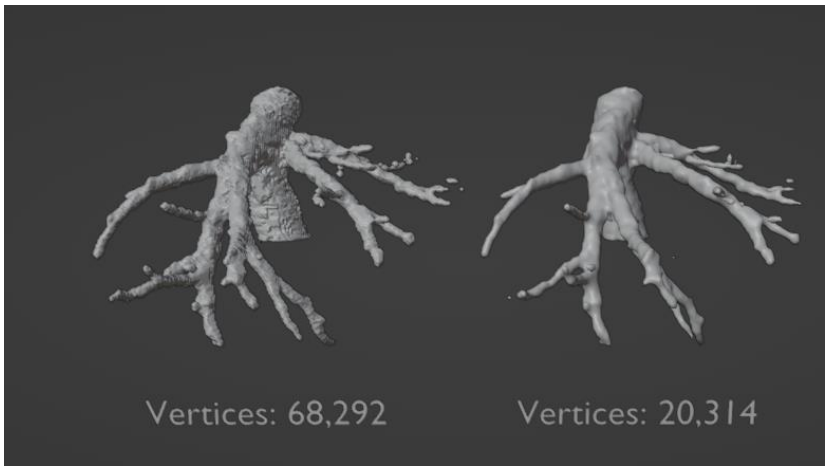


Figure 19 Result of applying the node structure to the vessel structure.

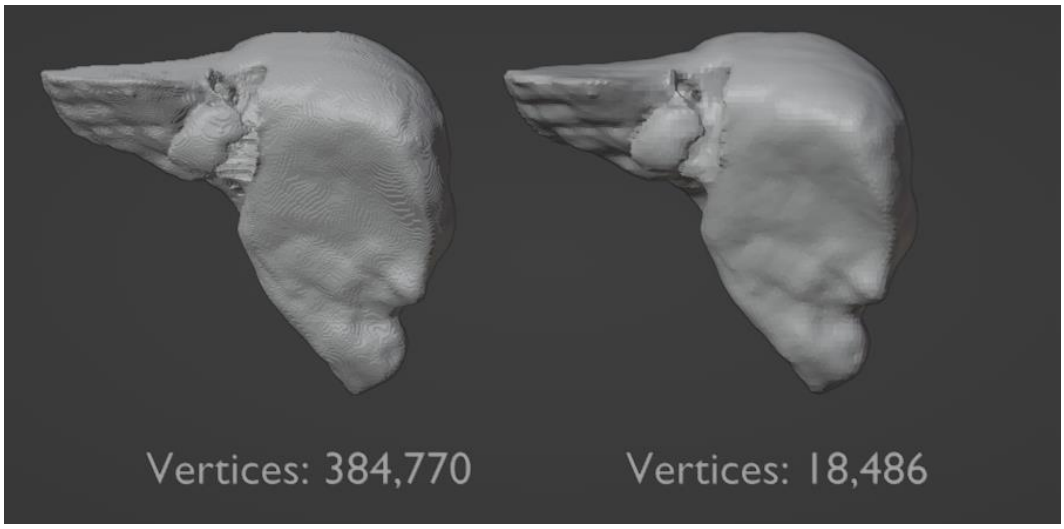


Figure 20 Result of applying the node structure to the liver.

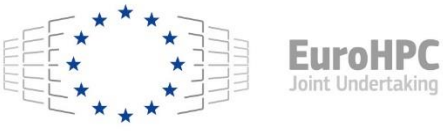
Conclusion

The current workflow described involves processing medical CT data using deep learning models, segmenting desired tissues, and generating 3D models for visualisation. The workflow includes steps such as automatic tissue segmentation, validation by medical doctors, post-processing, and exporting the models. The models are then viewed in augmented reality using Misterine Studio. To streamline and automate the workflow, several tasks and requirements are outlined. First, the functionality of the IT4Innovations AIAA extension workflow needs to be converted into a scripting command-line workflow. This involves importing necessary modules, creating scripts to handle input parameters, running inference on the CT data using the deep learning model, and saving the output masks. Similarly, the Misterine Studio UI workflow should be converted into a scripting command-line workflow. This requires analysing the UI workflows, writing scripts for each action, handling errors, and running the script with appropriate parameters. Additionally, the possibility of slicing the 3D model in any direction using a cutting object like a plane is desired. Methods for cutting the mesh and volume are provided for software such as Blender and 3D Slicer. These methods involve using tools like plane cuts, scissors, logical operators,

and cropping widgets to achieve the desired results. For materials, both seamless tileable textures and procedural materials can be used to enhance the appearance of the 3D models. Texture maps and procedural textures can be applied to the mesh and volume to create realistic materials and improve visual quality. Lastly, to optimise performance on mobile devices, the number of triangles in the 3D mesh should be limited to around 2 million. Techniques like decimation and geometry nodes can be employed to reduce the vertex count while preserving the overall shape and topology of the mesh. By addressing these tasks and requirements, the workflow can be streamlined, automated, and enhanced, enabling more efficient processing of CT data, better visualisation of 3D models, and improved performance in augmented reality applications.

References

1. 3D Slicer 2023. Available online: https://www.slicer.org/.
2. Blender 2023. Available online: https://www.blender.org/.
3. Misterine 2023. Available online: https://apps.apple.com/tm/developer/misterine-s-r-o/id1437793826, https://play.google.com/store/apps/details?id=com.misterine.misterineapp&hl=en_IN&gl=US .
4. Dynamic Modeler Surface Editing in 3D Slicer
   https://www.youtube.com/watch?v=F6fNMQTxD-4
   https://github.com/Slicer/SlicerSurfaceToolbox/blob/25999cefa2554b5fa4698b847716f75837070e33/DynamicModeler/Logic/vtkSlicerDynamicModelerPlaneCutTool.cxx
5. Clipping in 3D Slicer
   https://www.youtube.com/watch?v=x7Gpmj4DqOw
6. Image cropping using VTK
   https://kitware.github.io/vtk-js/examples/ImageCroppingWidget.html
   https://www.youtube.com/watch?v=GgPRsQYkGbc
   https://github.com/Slicer/Slicer/blob/f44849b014ecf9a245901239db64037556110970/Modules/Loadable/CropVolume/Logic/vtkSlicerCropVolumeLogic.cxx
7. 3D Slicer segment tools for cutting image volume
   https://www.youtube.com/watch?v=xZwyW6SaoM4
8. Cropping volume in 3D Slicer
   https://www.youtube.com/watch?v=xZwyW6SaoM4
   https://github.com/Slicer/Slicer/blob/9eb1f0862d8167fbb33265180046d0db98beb3e0/Modules/Loadable/Segmentations/EditorEffects/Python/SegmentEditorMaskVolumeEffect.py#L7



This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No. 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, the Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, the Netherlands, Belgium, Luxembourg, Slovakia, Norway, Turkey, Republic of North Macedonia, Iceland, Montenegro, and Serbia. This project has received funding from the Ministry of Education, Youth and Sports of the Czech Republic.