Project

***Due time**: Dec. 17th, 2025, 23:59*

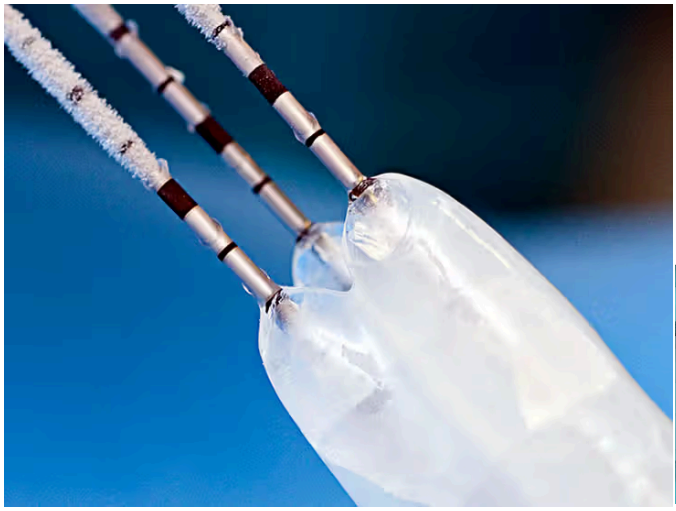
***Modalities**: A zip file containing your extension will have to be uploaded on Moodle, and the project will be presented during the last class session on Fri. Dec. 18th. Click on the "3D Slicer project" link on the main page to upload your zip file. The zip file shouldn't contain the given data, only your extension.*

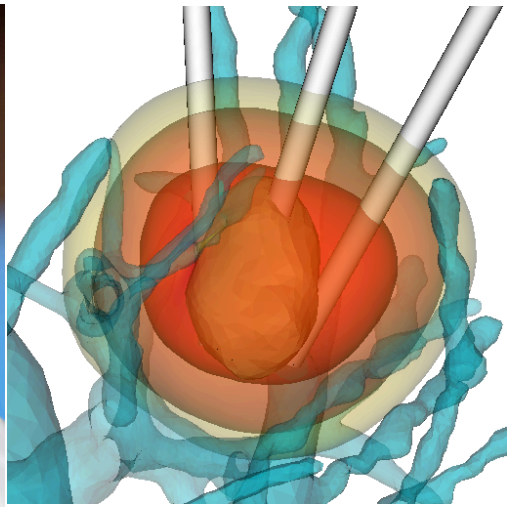
Please note that the project must be carried out in pairs of students.

Context:

Thermal ablation is a surgical procedure used in the treatment of abdominal cancer to precisely destroy the tumors without needing open surgery. It involves implanting one or several needles through the skin to deliver an extreme heat that will cause the necrosis of tumoral cells. This minimally invasive approach provides very good results with minimal invasiveness and allows for shorter hospital stays and recovery. Computer-assisted tools play a crucial role in planning and guiding the placement of these needles, ensuring safety and accuracy. Such technologies can optimize outcomes by reducing risks and improving the surgeon's precision. Finding a good placement for the needles is a complicated task for the clinician, as the needles mustn't damage sensitive structures such as vessels, and shouldn't be too close to each other.

Thermal ablation illustrations





Objective of the project:

The objective of the project is to compute the risk associated to a set of candidate needles in the liver of a patient, as well as the volume of destroyed tissues. In this project, we will use the following simplified hypotheses:

- the needles will be represented as cylinders between two points,
- the risk associated to a needle will be represented by the distance from its cylinder to the vessels,
- the volume of ablation associated to a needle will be represented as a sphere centered on a point located at 5mm from the tip of the needle,
- the total volume ablated will be represented as the union of all the ablation spheres.

The following dataset can be used for testing your program: Liver-Scene.zip (which was generated from the open source database 3D-IRCADb-01).

After extracting the compressed file, you will obtain multiple .vtk files, each one containing one model, and one .mrml file containing the 3D scene. In 3D slicer, opening only the .mrml file is sufficient to load the whole dataset.

This project will have to be entirely developed starting from the skeleton of application given as a basis: myNiceExtension.

?

You will first **rename** the extension, the module, and all the classes with a new title containing **your last name(s)** (to avoid confusion between projects when grading them). Be careful to update also the CMakeList files accordingly.

Tasks:

1) Needle and interaction

A needle will be represented by a cylinder of radius 1mm and length 15cm. In order to get cylinders with enough points, we will not use vtkCylinderSource. We will use vtkTubeFilter linked to a vtkLineSource (see vtkTubeFilter example in python). To avoid further issues later in the assignment, you will also apply a vtkTriangleFilter (see the vtkTriangleFilter class in the documentation) after the vtkTubeFilter to transform the resulting mesh into triangles instead of triangle strips.

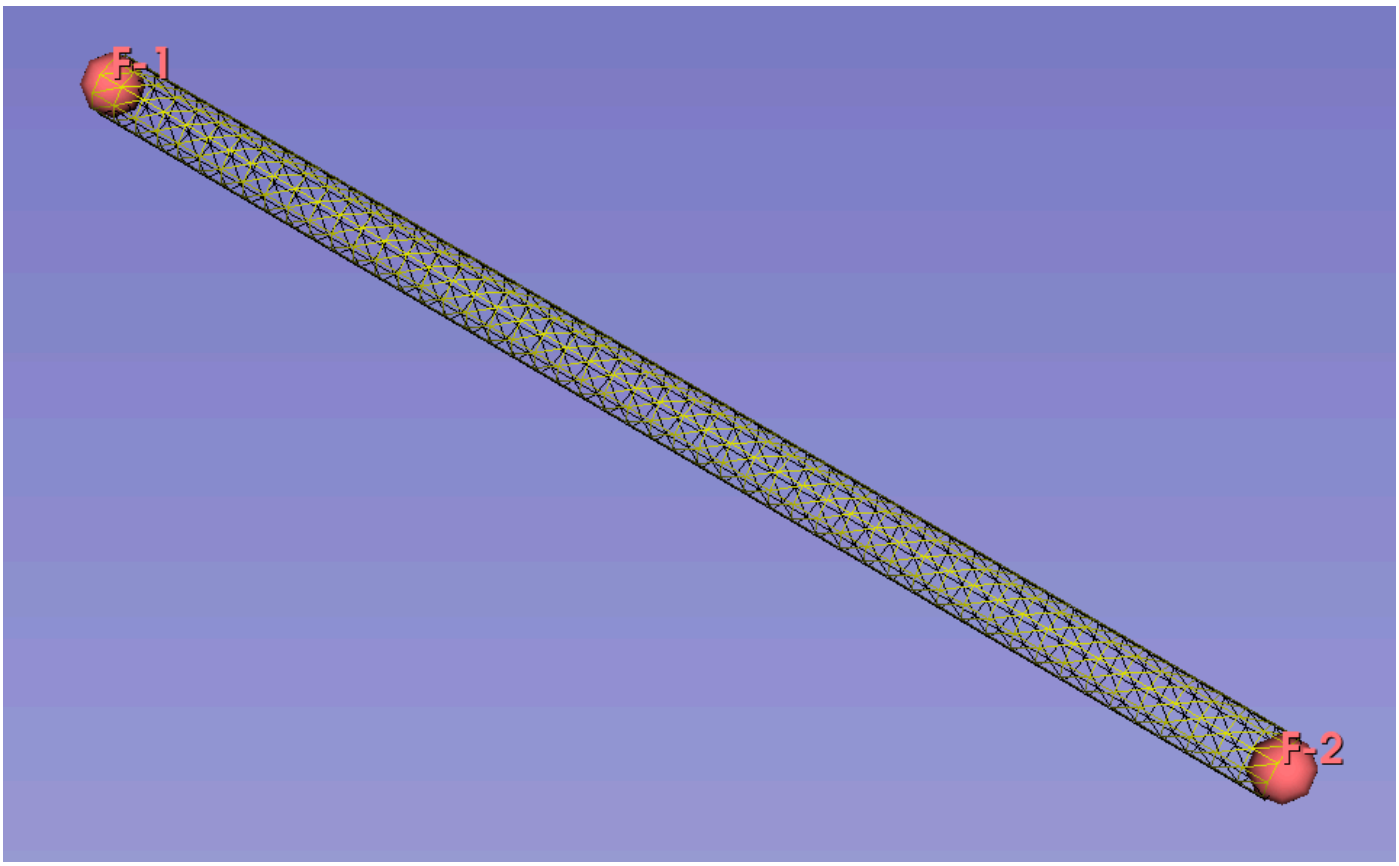
Each cylinder will be built between two distal points represented by two control points of a point list (markup). A point list will contain pairs of distal points to manipulate several cylinders: F-1 and F-2 for cylinder 1, F-3 and F-4 for cylinder 2, etc. We will work with a simplified hypothesis where the cylinders have a variable length depending on the position of their distal points.

The first task is to implement the creation of the cylinders, by designing a button and the associated callback functions. The cylinders will have to move accordingly when manipulating its distal points. Pressing the button should create as many cylinders (electrodes) as the number of targeted areas in the 3D scene. To avoid manually creating many fiducial points each time you want to test the module, it is recommended to create fiducial points automatically.

Note that it is not necessary to delete and recreate the cylinder when a point is moved, you can just update the original vtkLineSource from the new points positions. This requires to keep track of the original vtkLineSource as a member variable.

To help with the visualisation of the distal points (F-1 / F-2, F-3 / F-4, etc), you can increase their size (SetGlyphScale).

The result should look like this if the cylinder is displayed in wireframe:



2) Automatic needle placement

The interface should contain a button to automatically place the tips (F-1, F-3, etc) of the needles inside the tumor mesh. The second task is to implement this functionality. Each tip should be placed in the center of mass of the tumor.

When placed, it should still be possible to interact with the needle, especially to rotate it around its tip by moving the other distal point (F-2, F-4, etc.).

3) Computation of distances and display of result

We want to display the distance between the needles bodies and all the meshes representing vessels.

The third task is to implement a button and related callback function that finds the meshes of the corresponding sensitive structures in the MRML scene, computes the distance between the body of each needle and the structures, compute the minimum distance for each structure, and displays it in the interface.

For example, if needle #1 is at 2cm from one of the vessels meshes and 1.5 cm from another, while needle #2 is at a distance of 1cm and 3cm respectively form the same structures, then the interface should display minimal distances of 1cm for the first and 1.5cm for the second, which gives an idea of the risk of this combination of positions of the two electrodes.

Note that:

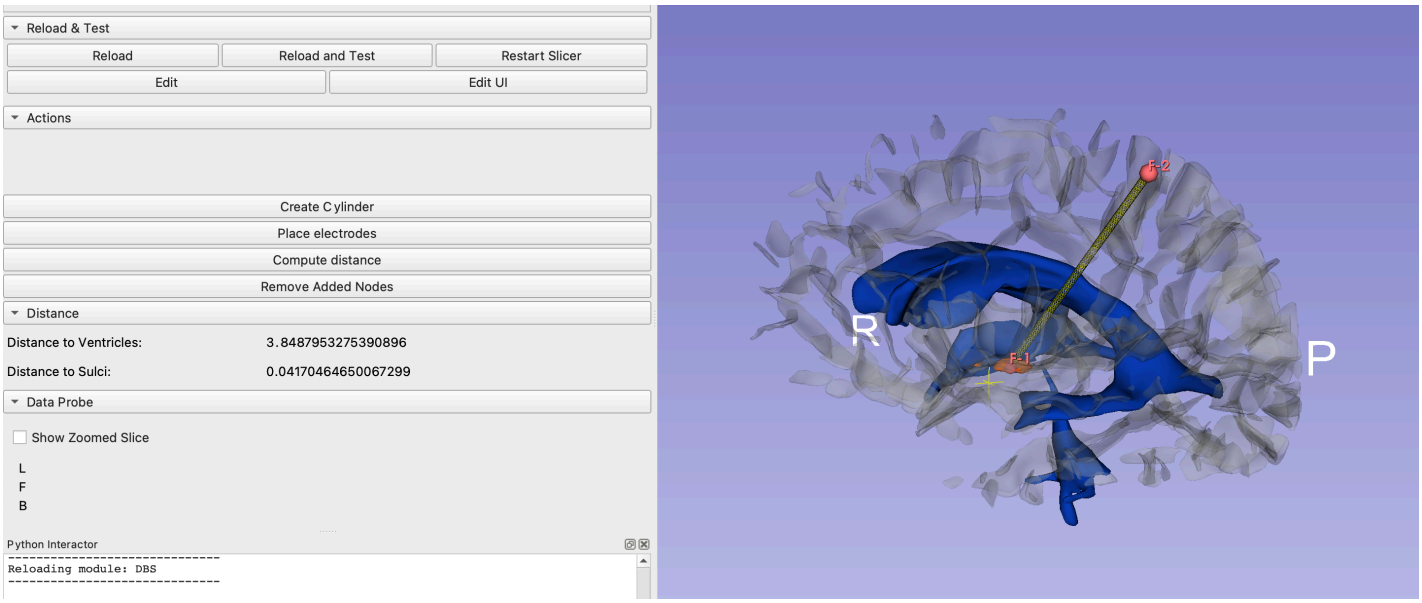
- the meshes in question can be found using their name
- to compute the distance between two polydata, you should use a vtk filter that computes the distance from each point of a first polydata to the other polydata. Then find the minimum of the generated array. If you experience problems using the filter, try swapping the inputs

4) Automatic update of the distance

The fourth task is to add a functionality to automatically update the display of the distances when one of the "free" points (F-2, F-4, etc.) is moved.

NB:

If you chose to display the distances in the python console, you may notice that the display is not very appealing, as it is constantly scrolling. Another option is to display the distances as labels in the widget panel and update them. This is a little bit more challenging, and requires to store a reference to the labels as member variables in the Logic so that you can access and update it from the Logic. For an even better arrangement of the buttons and labels, the different widgets can be placed in containers of type ctk.ctkCollapsibleButton, and the labels placed in containers of type qt.QGridLayout, like in the view below.



Note that the distances must be computed fast enough so that they should be displayed in interactive time when the electrodes are moved.

5) Automatic coloring according to the risk

The interface should change the color of the needle closest to the vessels to red.

6) Modeling the ablation

When a needle is added, the corresponding sphere representing the ablation should be created at the same time, and placed at 5mm from the internal tip of the needle. When the needle is moved, the position of the sphere should be updated accordingly.

7) Computing the volume ablated

In addition to the risk (distances) we want to display the volume ablated, as well as the efficiency: percentage of tumor covered by the ablation, and percentage of ablated volume that belongs to the tumor.

The best way to achieve this is to convert the meshes of the tumor and the ablation spheres into volumes of voxels, and then to:

- count the voxels belonging to the tumor to get the tumor volume
- count the voxels belonging to at least one sphere to get the ablated volume

- count the voxels belonging to the tumor and at least one sphere to get the volume of tumor ablated
- from these values, you can infer the volumes and percentages

Several strategies can be applied : you can use a filter to convert the meshes into volumes, or browse the entire image volume and check for each voxel if they belong to a mesh corresponding to the tumor or a sphere.

Once calculated, display the obtained volumes and efficiency values along the distances. Similarly to the distances, these values should be updated when the needles are moved.

Modifié le: lundi 17 novembre 2025, 18:17