SOFA Haptic Avatar Plugin

SOFA Plugin Manual

Version 1.0 - 2020-11-02 (Draft)

Author: Erik Pernod

1 - Overall description

1.1 - Haptic Avatar

Haptic Avatar is a haptic device for minimally invasive surgical training, e.g. laparoscopy, arthroscopy and thoracoscopy. It has four degrees of freedom, all with force feedback, and a form factor that enables multiple and adjustable portal placement suitable for both individual and team training.

It features a port-like part where a representation of a surgical tool can be inserted and retracted. The tool will be detected and identified at insertion, which opens up new and more realistic possibilities for training.

Here are a representation of some of the possible setup :

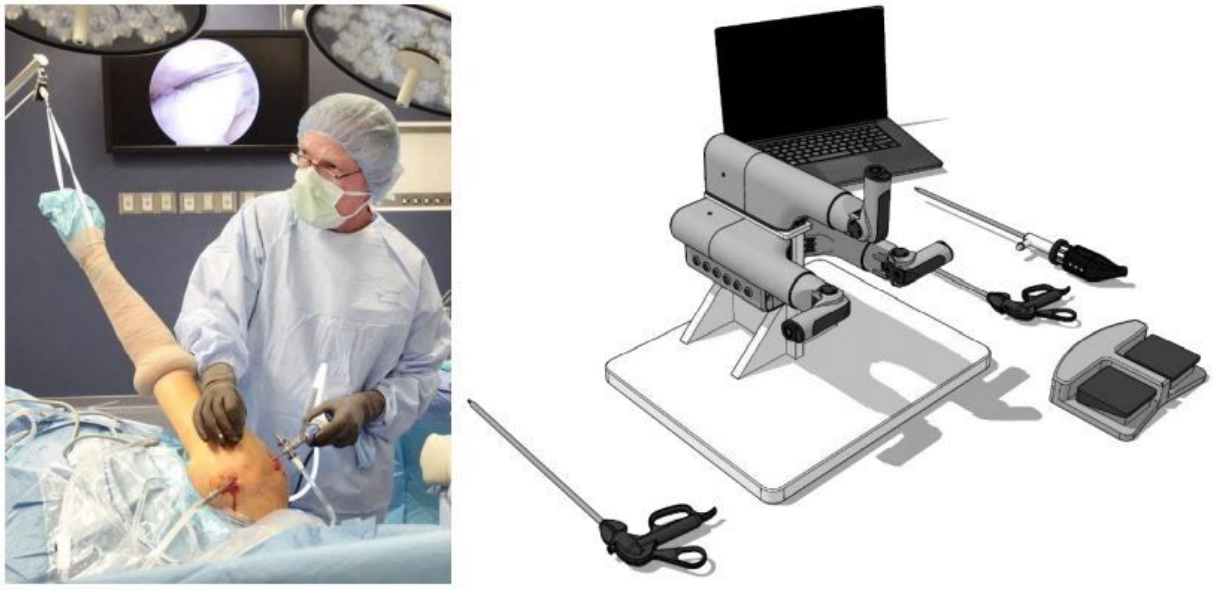


Figure 10. (Left) An arthroscopic shoulder procedure with the patient in lateral decubitus position. (Right) A simple corresponding setup for training.

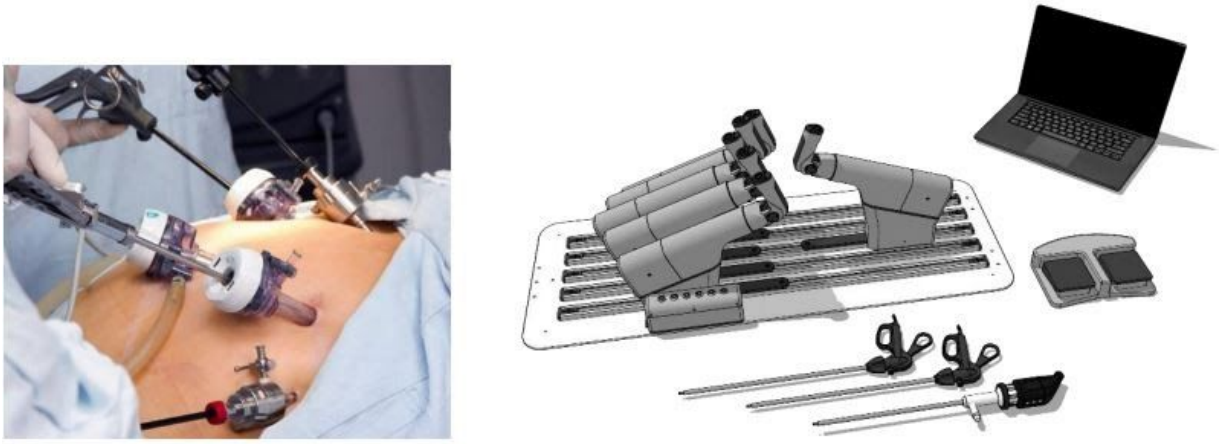


Figure 8. (Left) Picture of a bariatric surgery procedure. (Right) An example of a setup for laparoscopic general surgery.

For a full description of the device, please refer to the System Overview document.

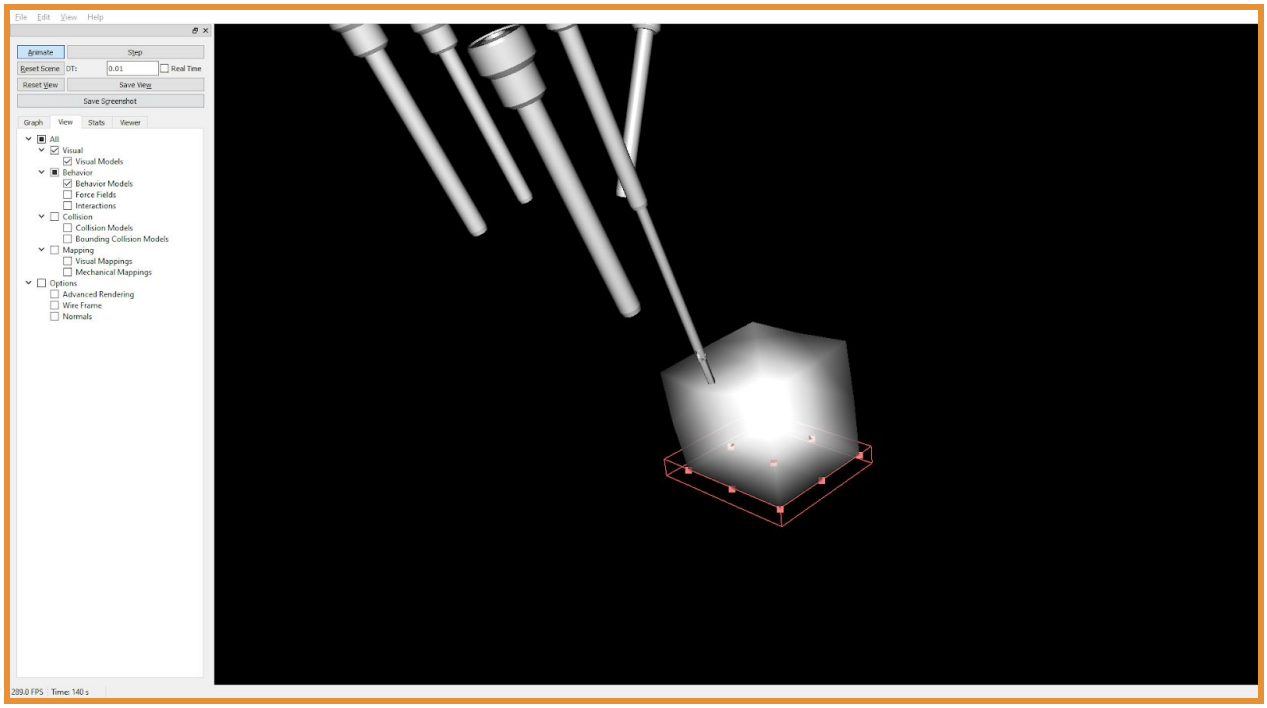
1.2 - SOFA

SOFA is an open-source framework for real-time interactive physics simulation, with an emphasis on soft body dynamics, robotics and medical simulation. Started in 2006, SOFA benefits today from a large international community made up of research centers and companies.

Thanks to its LGPL v2.1 open-source license (permissive and non-contaminating), SOFA is an innovation catalyst by making research efforts re-usable. SOFA is made up of a stable open-source core, providing state-of-the-art models and numerical methods, and many optional plugins (+150 open- and closed-source plugins) implemented specific features. The framework license and architecture therefore foster the development of prototypes and products under any commercial license.

1.3 - SOFA Haptic Avatar Plugin

This manual will describe how Haptic Avatar devices are integrated and simulated inside SOFA through this plugin.



For a full description of the device functionalities and communication protocol, please refer to the Haptic Avatar Reference Manual.

2 - Plugin content

2.1 - Repository Folder

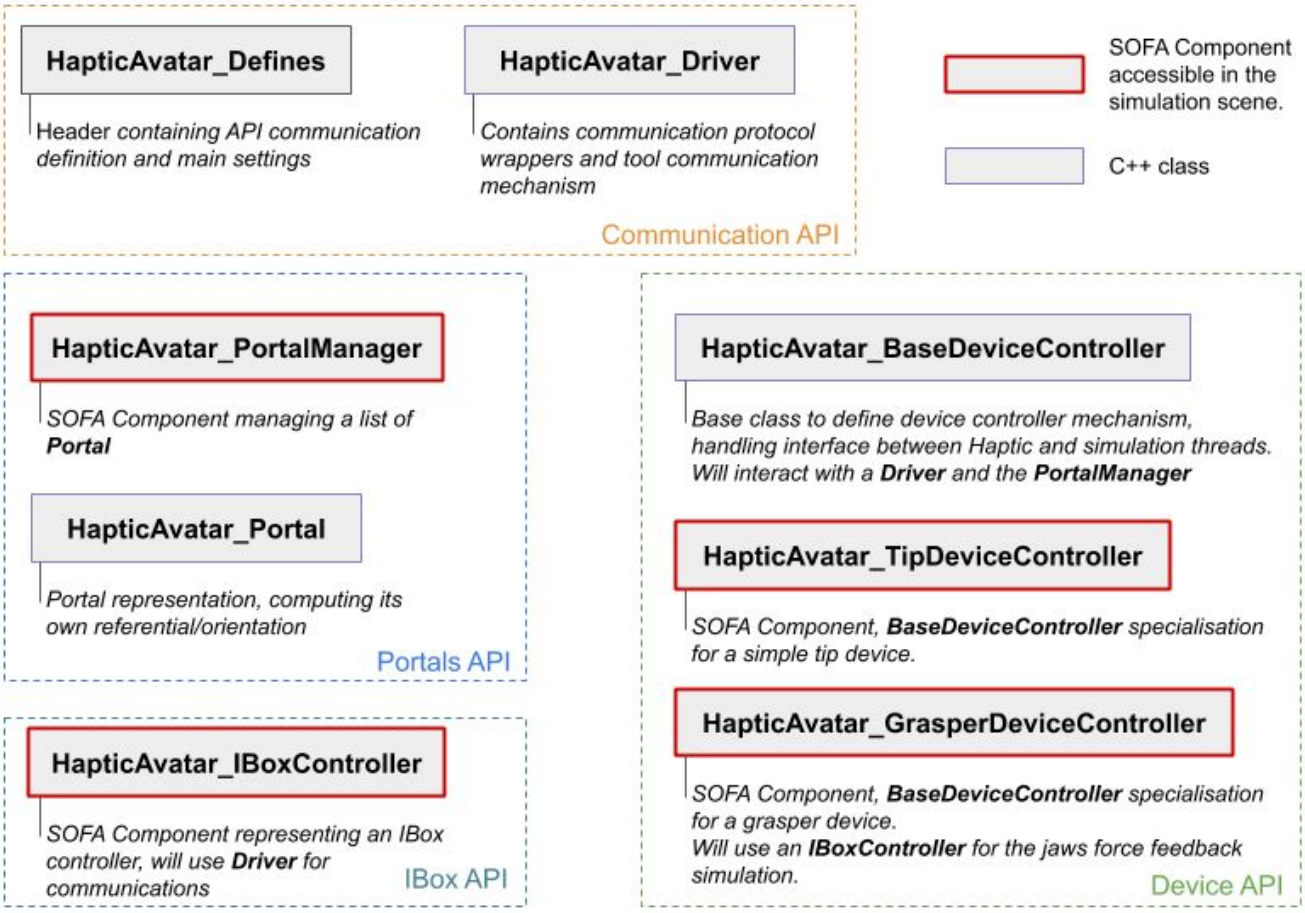
This SOFA Haptic Avatar plugin is available on GitHub at this URL:
https://github.com/epernod/SofaHapticAvatar

This repository contains:

- **examples:** A list demo examples
- **src:** the plugin c++ code classes
- **doc:** this documentation as well as device manual reference

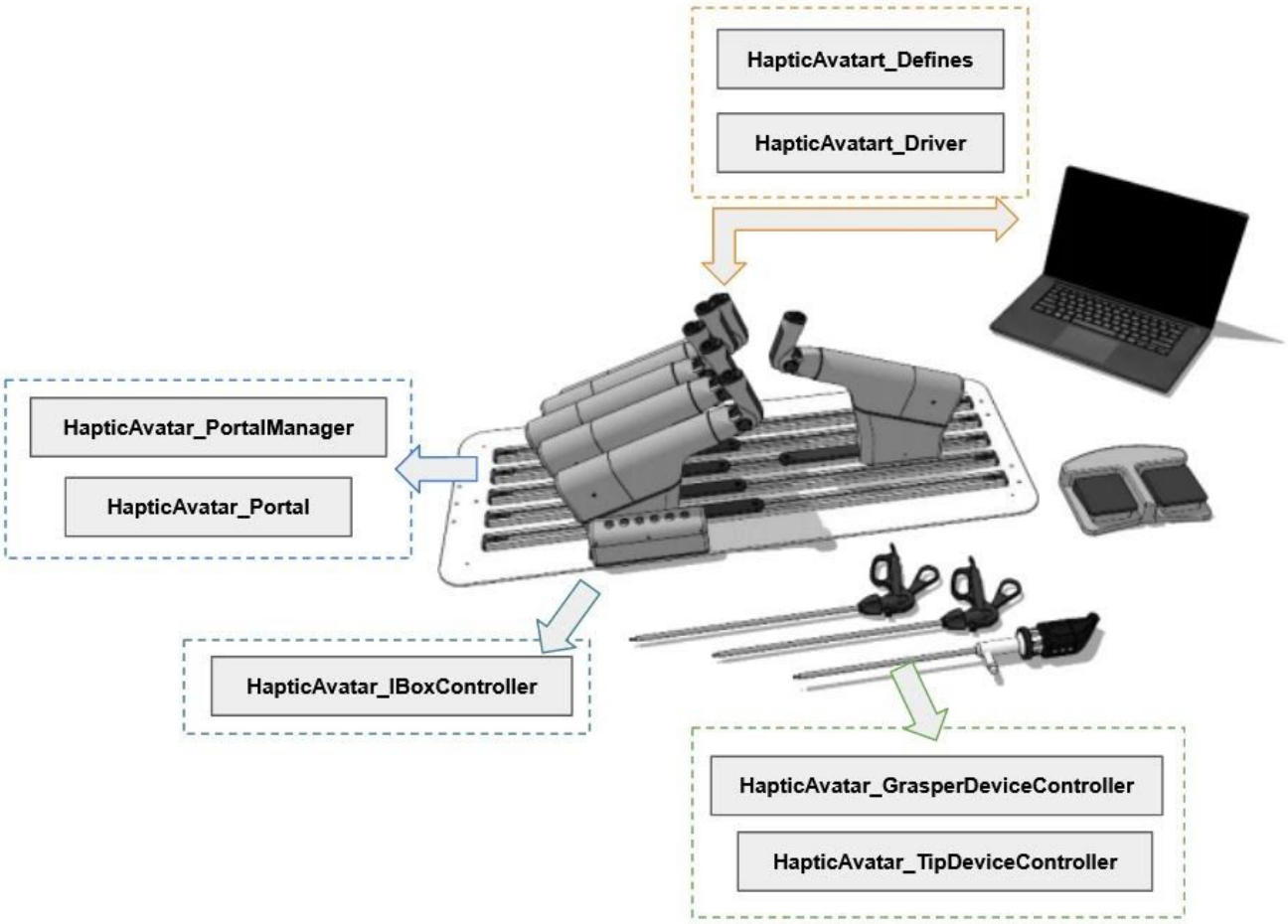
2.2 - Code architecture

Here is the list of classes available in this plugin with a small description



2.3 - SW vs HW

Here is a simplified representation of each C++ class scope and use compared to the Haptic Avatar devices.



2.4 - List of examples

TO be completed
HAvatar_tool_motion
HAvatar_tool_rigidCactus
HAvatar_tool_rigidCube

HAvatar_deformableCube

HAvatar_grasper_motion
HAvatar_grasper_motion2

HAvatar_grasper_rigidCube

3 - Plugin installation

3.1 - SOFA installation

SOFA physical engine can be downloaded here and installation procedure can be found here.
The plugin has been developed with version XXX of SOFA and is compatible with XXX

3.2 - Plugin installation

This SOFA Haptic Avatar plugin is available on GitHub at this URL:
https://github.com/epernod/SofaHapticAvatar And should be downloaded on the computer.

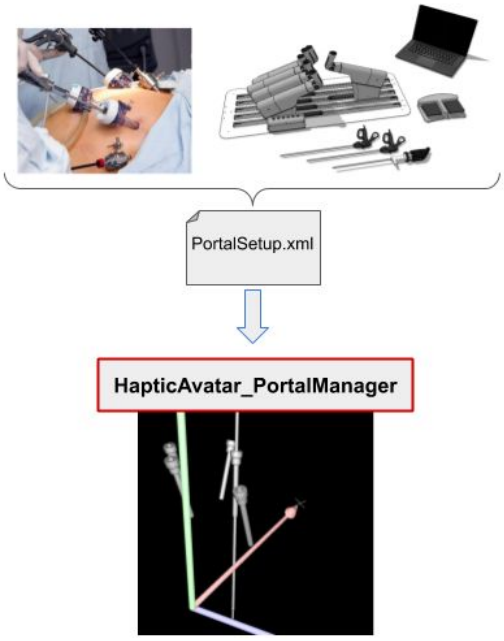
During SOFA CMake configuration:

1. Set the variable SOFA_EXTERNAL_DIRECTORIES to the location where the plugin has been downloaded like for example: “C:/projects/sofa_plugins”.
2. Configure CMake once. The option PLUGIN_SOFAHAPTICAVATAR will appear.
3. Check the option, configure again and generate the solution.
4. Build the project solution like explain in section **3.1**

4 - Haptic Simulation mechanism

4.1 - SOFA Scene creation

The portals need to be placed depending on the setup wanted, this setup is described through an XML file which need to be loaded into the HapticAvatar_PortalManager component like this:
<HapticAvatar_PortalManager name="portalMgr" configFilename="./config/PortalSetup.xml" />



The XML file looks like:

```
<?xml version="1.0" encoding="utf-8" ?>
  <Procedure Name ="Cholecystectomy">
  <Portals>
    <Portal Number ="1" >
        <PortalSettings Rail ="2" RailPos ="0" FlipAngle ="0" TiltAngle ="20" ComPort ="COM4"/>
    </Portal>
    <Portal Number ="2" >
        <PortalSettings Rail ="-1" RailPos ="55" FlipAngle ="0" TiltAngle ="20" ComPort ="COM7"/>
    </Portal>
    <Portal Number ="3" >
        <PortalSettings Rail ="0" RailPos ="0" FlipAngle ="0" TiltAngle ="20" ComPort ="COM5"/>
    </Portal>
    <Portal Number ="4" >
        <PortalSettings Rail ="0" RailPos ="0" FlipAngle ="0" TiltAngle ="0" ComPort ="COM3"/>
    </Portal>
    <Portal Number ="5" >
        <PortalSettings Rail ="0" RailPos ="100" FlipAngle ="180" TiltAngle ="20" ComPort ="COM6"/>
```

```
    </Portal>
  </Portals>
</Procedure>
```

Those information define local transformation of the different portals. The PortalManager has its own SOFA Data field ***"position"*** to set the global 3D position of the set of portals inside the SOFA 3D scene.

Then the PortalManager need to be linked to any HapticAvatar_DeviceController like this:
<HapticAvatar_DeviceController name="HADevice" portName="//./COM3" portalManager="@portalMgr"/>

Note that if a HapticAvatar_IBoxController is also needed, it should be added in the same way in the scene and linked to the DeviceController. The final result should look like:
<HapticAvatar_PortalManager name="portalMgr" configFilename="./config/PortalSetup.xml" />
<HapticAvatar_DeviceController name="HADevice" portName="//./COM3" portalManager="@portalMgr" />
<HapticAvatar_IBoxController name="HAIBox" portName="//./COM5" />

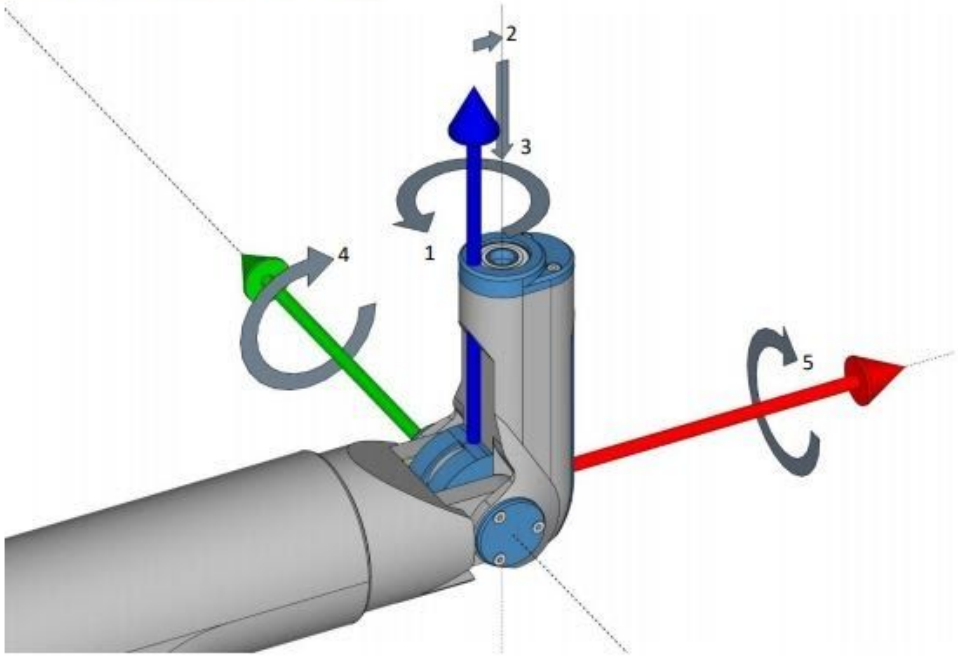
4.2 - HapticAvatar Device mechanism

The scheme below shows the possible tool motion and the information which can be retrieved.

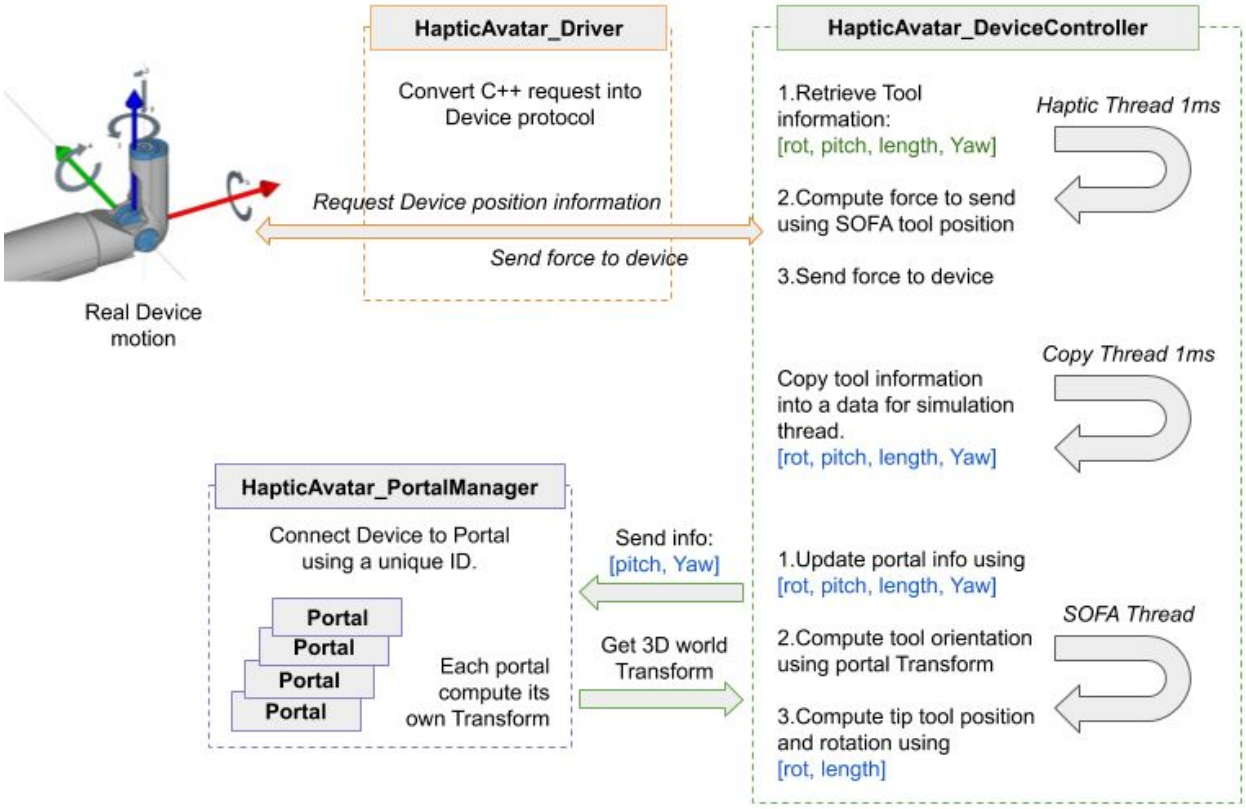
To create the correct tool pose from the angles and length provided from the Haptic Device, in relation to the LCS, the following steps are taken. The assumption here is that the tool is modeled with its shaft tip standing on the origin and extending along the positive z-axis.

1. Rotate the tool ROT-angle counter-clockwise around the z-axis.
2. Translate the tool 8.8mm in the positive x-axis direction (see note below).
3. Translate the tool Z-length along the z-axis. Please note that as the tool is inserted, z decreases (becomes more and more negative).
4. Rotate the tool PITCH-angle counter-clockwise around the y-axis.
5. Rotate the tool YAW-angle counter-clockwise around the x-axis.

The figure below illustrates these steps



Here is a diagram showing how those information are requested from the Device controller and passed to the SOFA context.



A first thread is used to communicate with the Haptic device. This is the Haptic thread running at 1000Hz. It is used to retrieve the tool information and send the force feedback using SOFA mechanism which will be described in the next section. Then a second thread running as well at high frequency is used to copy the tool information into a data set which will be used by the simulation thread. This step is a security to avoid write vs read access issues.
Finally the main thread is the simulation thread running at SOFA simulation speed (low frequency). This thread uses the copied information to update the portal information which will compute its own 3D Transform in the SOFA world. Using this Transform and the tool rotation and insertion length, the final position of the tool in SOFA is computed.

4.3 - SOFA Device representation

The real device is represented virtually in SOFA using an ArticulatedSystemMapping to define the different articulation of the device. Each articulation represents either a single rotation or a translation in one direction.

Thus the device is decomposed into 3 articulations:

- One at the trocar position which allows 3 differents rotation (ROT, Yaw and Pitch)
- A second articulation that represents the tool translation

- A last articulation in the case of grasper where the 2 yaws can open.

Scheme to update


```
<Node name="articulationCenters">
    <Node name="articulationCenter1">
        <ArticulationCenter parentIndex="0" childIndex="1" posOnParent="0 0 0" posOnChild="0 0 0" articulationProcess="0" />
        <Node name="articulations">
            <Articulation translation="0" rotation="1" rotationAxis="1 0 0" articulationIndex="0" />
        </Node>
        <Node name="articulations">
            <Articulation translation="0" rotation="1" rotationAxis="0 1 0" articulationIndex="1" />
        </Node>
        <Node name="articulations">
            <Articulation translation="0" rotation="1" rotationAxis="0 0 1" articulationIndex="2" />
        </Node>
    </Node>

    <Node name="articulationCenter2">
        <ArticulationCenter parentIndex="1" childIndex="2" posOnParent="0 0 0" posOnChild="0 0 0" articulationProcess="0" />
        <Node name="articulations">
            <Articulation translation="1" rotation="0" rotationAxis="0 1 0" articulationIndex="3" />
        </Node>
    </Node>


    <Node name="articulationCenter3">
        <ArticulationCenter parentIndex="2" childIndex="3" posOnParent="0 0 0" posOnChild="0 0 0" articulationProcess="0" />
        <Node name="articulations">
            <Articulation translation="0" rotation="1" rotationAxis="0 0 1" articulationIndex="4" />
        </Node>
    </Node>

    <Node name="articulationCenter4">
        <ArticulationCenter parentIndex="2" childIndex="4" posOnParent="0 0 0" posOnChild="0 0 0" articulationProcess="0" />
        <Node name="articulations">
            <Articulation translation="0" rotation="1" rotationAxis="0 0 1" articulationIndex="5" />
        </Node>
    </Node>
</Node>
```

A 1D mechanical Object is used to store the float value of each articulation….

4.4 - Force Feedback computation

The force feedback computation is done using the LCPForceFeedback component .
Documentation to come from meeting