

SOFA – Unity3D Asset

V23.12: [Online documentation](#)

Contact: contact@infinyTech3D.com

GitHub: <https://github.com/InfinyTech3D/SofaUnity>



Version 1.0 - 2024-04-30

Contents

1. Project status and summary.....	Erreur ! Signet non défini.
1.1 Introduction to Medical Context.....	Erreur ! Signet non défini.
1.2 Robotic Percutaneous Coronary Intervention	Erreur ! Signet non défini.
2. Project objectives	Erreur ! Signet non défini.
3. Intervention modelling.....	7
3.1 Anatomy modelling	Erreur ! Signet non défini.
3.2 Anatomy simulation	Erreur ! Signet non défini.
3.3 Tool modelling.....	Erreur ! Signet non défini.
4. Scenario simulations results.....	Erreur ! Signet non défini.
4.1 Simulation architecture.....	Erreur ! Signet non défini.
4.2 Benchmarks scenario.....	Erreur ! Signet non défini.
4.3 Full Demo simulations	Erreur ! Signet non défini.
5. Simulation improvements	Erreur ! Signet non défini.
5.1 Work on simulation speedup	Erreur ! Signet non défini.
5.2 Tools mechanical behavior.....	Erreur ! Signet non défini.
5.3 Fluoroscopic rendering.....	Erreur ! Signet non défini.
6. Validation Process	Erreur ! Signet non défini.
Confronting simulation to real Data for enhancement.....	Erreur ! Signet non défini.
7. Software interface on Unity3D.....	Erreur ! Signet non défini.
8. Conclusion & Future Steps	Erreur ! Signet non défini.
9. Annexes	Erreur ! Signet non défini.
Technical tasks description.....	Erreur ! Signet non défini.
Packages description	Erreur ! Signet non défini.
Reference size scheme for 3D printing	Erreur ! Signet non défini.

1. Introduction

1.1 - What is SofaAPAPI-Unity3D ?

The goal of SofaAPAPI-Unity3D is to combine performance and accuracy of scientific multi-physics simulation with the Virtual Reality world.

More concretely, it is a combination of two software programs working together.

On one hand, from it's full name **SofaAdvancedPhysicsAPI**, is a C++ library.

It aims to optimize and wrap the concepts of the SOFA framework in a single API with several levels of integration. On the other hand, **SofaUnity** is a **Unity3D C#** asset integrating this API to propose **SOFA** as a physics engine complementary to **Unity3D PhysX**.

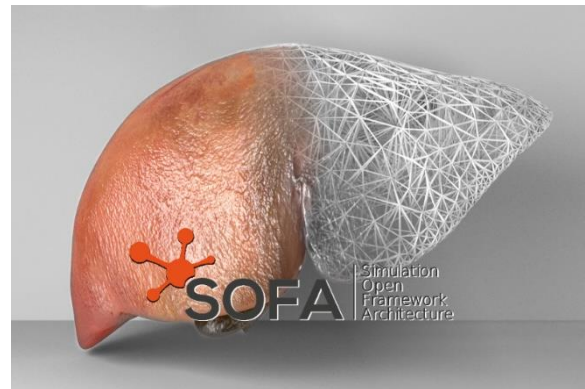


1.2 - But what is SOFA ?

SOFA, meaning Simulation Open Framework Architecture, is an efficient framework dedicated to research, prototyping and development of physics-based simulations using scientific approaches and concepts such as Finite Element Methods, constraint and numerical methods for ODE resolution.

For full description visit:

www.sofa-framework.org



1.3 - What can I do with it ?

SofaAPAPI-Unity3D can be used in Unity3D to simulate solid and deformable mechanics as well as Fluid and Thermodynamics.

It also support multiple haptic devices interaction with continuous constraint resolution.

Check the section [Features Available](#) for more details.

1.4 - How to get it ?

The core of the SOFA integration inside Unity3D is available for free on the Unity Asset Store and is also available on [Github](#). Several additional features such as haptic integration, medical imaging or ready to use surgical scenario can be obtained on demand using the form in the [Request software section](#). Those assets are available under different licensing terms depending on your use. See the [License section](#).

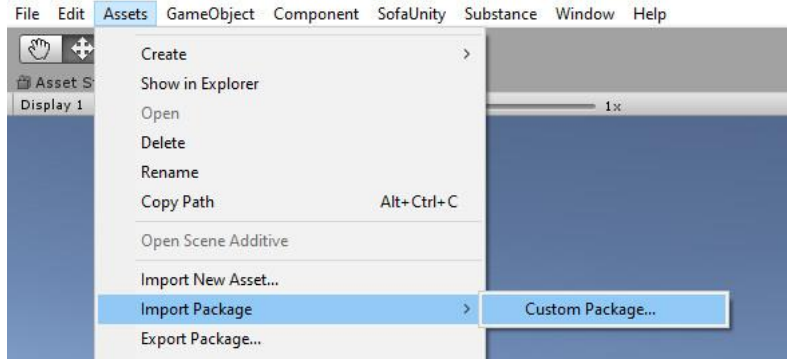
For more details, please contact: contact@infinytech3d.com

2. Getting started

2.1 – How to install

To fully install the asset, several steps are needed and depends on the version you downloaded. you will need to install first the package using the Unity asset store or download the SofaUnity package from GitHub, unzip the Unity3D package and follow those steps:

Downloaded from Unity Asset Store	Downloaded from Github project page
1. From Unity3D, open your project or create a new 3D project. Use normal, URP or HDRP project depending on the package version you requested.	
2. In Unity3D Editor, use Assets -> Import Package -> Custom Package and load the .unitypackage .	

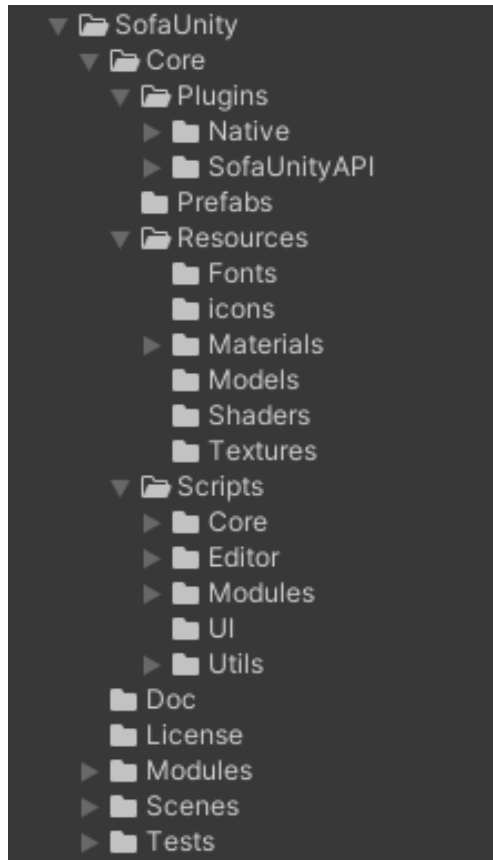


The screenshot shows the Unity3D Editor interface. The 'Assets' menu is open, and the 'Import Package' option is selected. A sub-menu is displayed, showing 'Custom Package...' as the selected option. The background shows the Unity Hierarchy and Hierarchy panels.

<p>3. From the Github project page, download the SOFA dll files from SofaUnity v23.12 windows dll.zip and unzip the folder into SofaUnity\Core\Plugins\Native\x64</p>	<p>3. Nothing to do: The SOFA dll files are already shipped in this asset version.</p>
<p>4. Only for specific modules: Once the package has been fully loaded, place the license file inside {your Unity3D project}/Assets/SofaUnity/License/. Replace the existing file if needed.</p>	
<p>5. Finally, check the config file: <i>{your Unity3D project}/Assets/SofaUnity/Plugins/Native/x64/sofa.ini</i></p> <p>It should contain 2 lines leading to your project path. It should looks like this: <i>SHARE_DIR=C:/projects/Unity/SofaUnity_default/Assets/SofaUnity/scenes/SofaScenes</i> <i>EXAMPLES_DIR=C:/projects/Unity/SofaUnity_default/Assets/SofaUnity/scenes/SofaScenes</i></p>	
<p>6. Change those paths if you want to use another directory where you store your SOFA scenes and meshes.</p>	

2.2 - Package content

Once ready to be used the SofaUnity asset folder should look like this:



- **Core:** The main content of this asset with C# scripts, SOFA integration API scripts & dll as well as resources for the examples.
 - **Plugins**
 - **Native:** SOFA x64 .dll files folder
 - **SofaUnityAPI:** API scripts interfacing with Sofa library
 - **Prefabs:** SOFA-Unity prefabs used in the scenes
 - **Resources:** Unity materials, textures, shaders and images
- **Scripts:** all asset classes in C#
 - **Core:** Class to represent SOFA objects using Unity3D GameObject
 - **Editor:** Class to implement Objects Unity Editor inspectors
 - **Modules:** Specific classes inheriting from Core classes
 - **UI:** 3D UI controlling SOFA player in game.
 - **Utils:** Scripts of useful components/algorithms.
- **License:** Folder to place license file for modules
- **Modules:** Specific features such as Haptic support, Xray, Ultrasound, etc. available as additional asset embedding SOFA plugin.
- **Scenes:** Folder containing several examples scenes:
 - **Benchmarks:** Some scene for speed/integration benchmarks
 - **Examples:** Several examples of SOFA capabilities
 - **Demos:** More complex scenes using SOFA features
 - **Tests:** Folder with scripts and reference of the scenes results

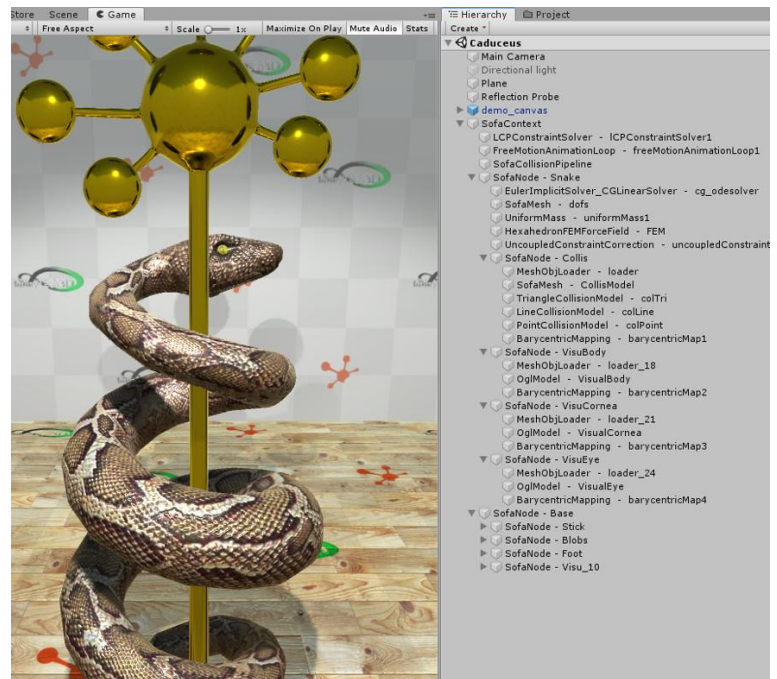
7. Main principles

To continue in this section, it is strongly recommended to have basic knowledge of Unity3D Editor and SOFA physics Engine.

- Full documentation and tutorials for Unity3D can be found here: [Unity Manual](#)
- Same for SOFA can be found here: [SOFA doc](#)
Do not hesitate to contact the SOFA consortium to get help on where to start and get information regarding trainings.

The asset allows you to run SOFA simulation in background while your Unity3D project is running. Basically, at each Update of the Unity3D application, a simulation step will be requested in SOFA. Note that the execution is performed asynchronously and thus the time step used in your physical simulation may be different to the Unity update frequency.

In more details, through a hierarchy of C# classes relying on the Unity3D monobehaviour mechanism, the asset provides a set of Unity3D scripts and GameObjects representing the main SOFA components. Like any GameObject, they can be edited using the Inspector view and added to the SOFA simulation scene graph through the Hierarchy view. See Object Creation of Scene Graph Edition for more details.

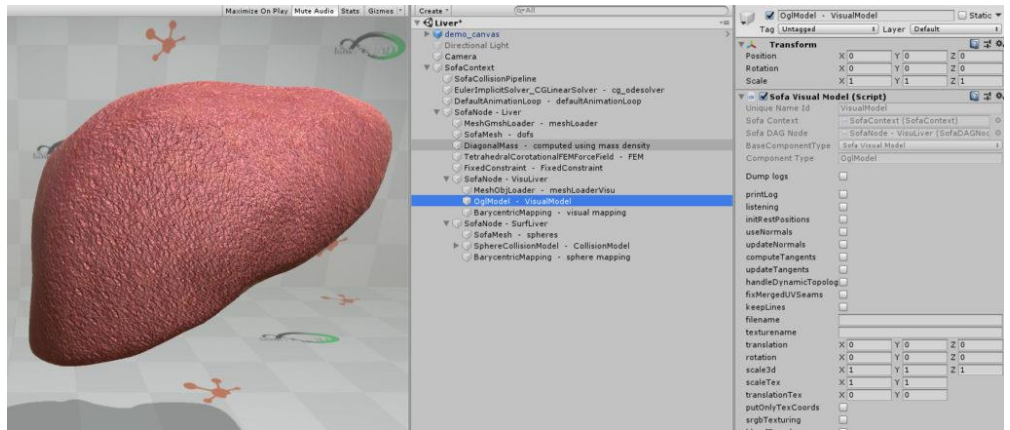


SofaComponent specialization

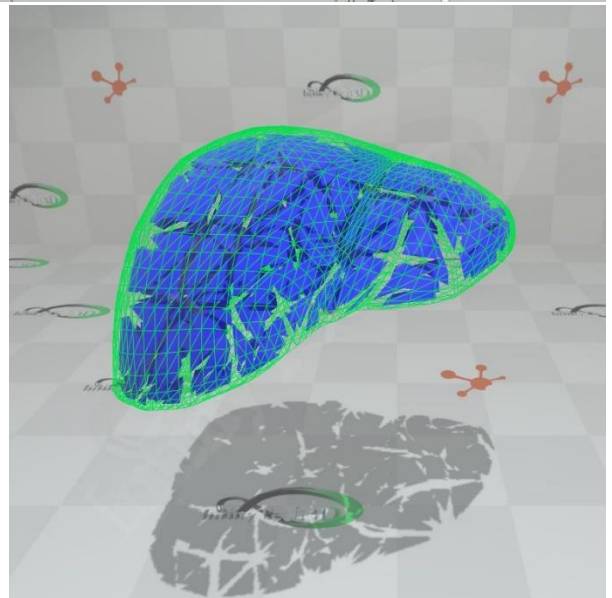
As explain earlier in the [SOFA Scene Parsing section](#). Several categories of component from **SOFA** are mapped as **Unity3D** specialized **GameObject**. The others are created as **SofaComponent** generic **GameObject**.

Those specialized implementation allows for example to see the multi-model representation of a 3D object. In the Liver scene it is possible to see the visual

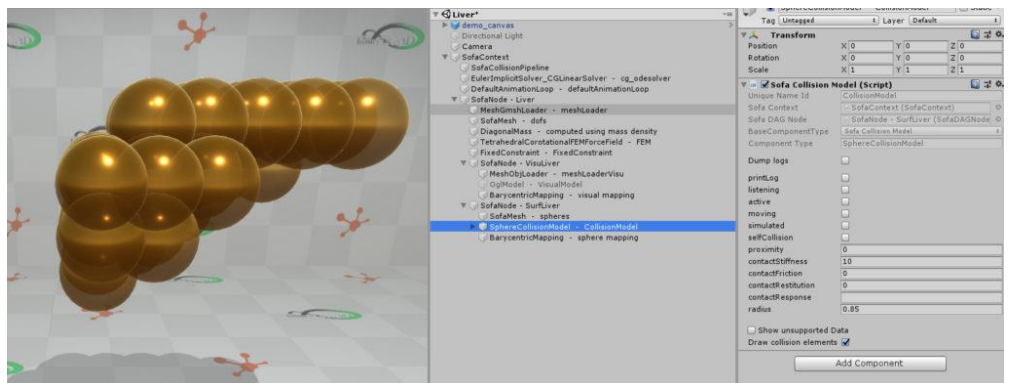
Classic visual rendering of the object using MeshFilter inside the component SofaVisualModel.



FEM element can be displayed inside the SofaFEMForceField by activating the Mesh Renderer. Note that it can slow down the simulation as for example the tetrahedron in this case need to be updated at each frame.



Collision model can also be displayed in the SofaCollisionModel by activating the option **Draw collision elements**. Note that same as for FEM, this can slow down the simulation as the element need to be updated at each frame.



SEVERAL API LEVELS

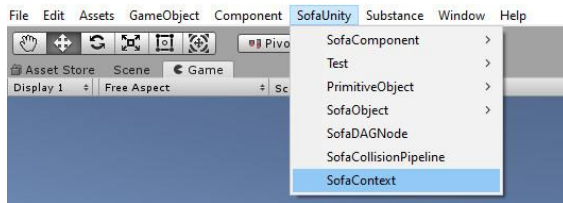
SOFA gather many different physics principles. It's simulation scene graph can quickly be complex. Therefore several level of integration are possible to create SOFA simulation using SofaAPAPI-Unity3D:

- The simplest way is by loading existing SOFA simulation scene (*.scn). See the section: [SOFA Scene Parsing](#)
- By creating high level SOFA-Unity objects. See the section: [Object Creation](#)
- Or by creating SOFA component representation directly inside the Unity3D scene graph. See the section: [Graph Edition](#)

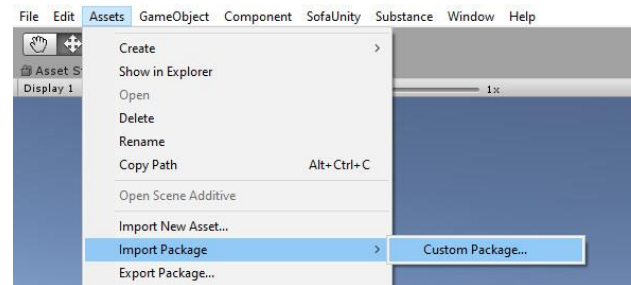
3.1 – SOFA Scene Parsing

To be able to use **SOFA** physical engine inside **Unity3D**, the first step, whatever API method is used, is to create the object: **SofaContext**. This is a **Unity3D** GameObject representing the **SOFA** simulation root object which will control the simulation in background. All other object to be simulated by **SOFA** will be child of this Root object.

This object can be created either using the top menu inside **SofaUnity** or directly by right clicking in the hierarchy panel.



Access to SofaContext from Top Menu

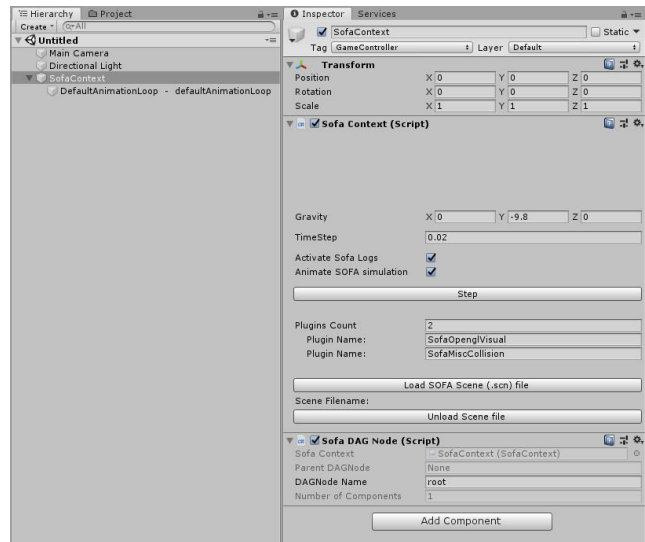


Access to SofaContext from Hierarchy panel

The inspector of the **SofaContext** **GameObject** allows to change the gravity in the scene as well as the simulation timestep. It also allows to activate the message handler in order to get **SOFA** logs (info, warning and errors) and disable the simulation if manual control is wanted using the button step.

Note that a root **SOFADAGNode** as well as a **GameObject** defaultAnimationLoop will be automatically created.

See next section for more details.



SOFA components to SofaUnity GameObjects

Since the version 1.0 of SofaAPAPI-Unity3D several components from **SOFA** are mapped as **Unity3D** specialized **GameObject**. The others are created as **SofaComponent** generic **GameObject**. The generic implementation of **SofaComponent** only allows to interact with the **SOFA** component parameters. Those parameters correspond to the **SOFA Data** of a component. Whereas the component specialization allow to add more complexe behavior. Like for example the display of FEM or collision model. For more details, check section [Scene Graph edition](#)

Here is the list of **Unity3D** specific **GameObject** used to define **SOFA** simulation scene.component categories particularly handled in :

SofaAnimationLoop

This **GameObject** correspond to **SOFA AnimationLoop**. This is a key component in charge of ruling the simulation and the mechanical resolution system.

See the corresponding documentation on **SOFA** website: [AnimationLoop](#)

SofaCollisionModel

This **GameObject** correspond to type of collision element used in **SOFA** to compute primitive intersection. It is usually either Spheres, Triangle, Edges or Points. This component is directly linked to the **SofaMesh**.

See the corresponding documentation on **SOFA** website: [IntersectionMethod](#)

SofaConstraint

This GameObject will describe some constraint applied to the current system. It is only there to help describing the simulation scene. It could be either projective or lagrange constraints. check more information regarding constraint on **SOFA** webiste: [Projective Constraint](#)

SofaFEMForceField

his GameObject correspond to **SOFA AnimationLoop**. This is a key component in charge of ruling the simulation and the mechanical resolution system.

See the corresponding documentation on **SOFA** webiste: [AnimationLoop](#)

SofaLoader

This GameObject is for the moment only used to display the type of mesh used. Several formats are handled in **SOFA**. The more known are: obj, vtk, stl or gmsh.

SofaMass

This GameObject correspond to the Mass component applied to this mesh. This component can be used to change the total mass of the object or its mass density.

More information regarding Mass can be found on **SOFA** webiste: [Mass](#)

SofaMechanicalMapping

This GameObject is used to inform if a mechanical mapping is used between different mesh object. The link to the input and ouput mesh will be displayed.

This is a particular mechanism of **SOFA** which is explained into details here: [mappings](#)

SofaMesh

This GameObject is one of the main component. It corresponds to the mesh used in used **SOFA**. This component will retrieve the number of points as well as their positions in 3D space but also the complete topology of the object.

This GameObject is the main container used to fill MeshFilter or other algorithms. More information regarding Topology can be found on **SOFA** webiste: [topology](#)

SofaSolver

This GameObject is used to inform which type of integration scheme is used as well as the linear solver used to resolve it.

More information regarding the mechanical system resolution can be fond on **SOFA** website in the [Integration Schemes section](#) and [Linear Solvers section](#)

SofaVisualModel

This GameObject is used to map **SOFA** output position from an OglModel/VisualModel to **Unity3D** rendering. A MeshFilter and a Mesh renderer will be created and linked to **SOFA** side.

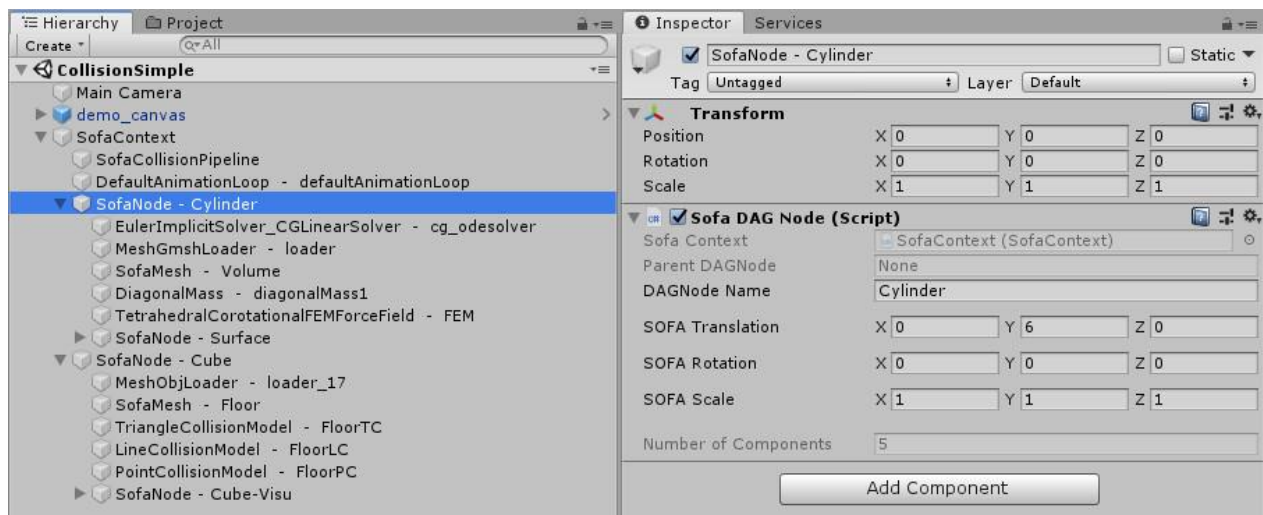
SofaDAGNode scope

SOFA simulation is using a *Directed Acyclic Graph* concept to design hierarchy structures, similar to what can be seen in **Unity3D** Hierarchy panel. The GameObjects SofaDAGNode are used to retrieve this scene Node architecture.

It is highly advise to be familiar with the **SOFA** scene graph structure [described here](#).

For example in the scene **CollisionSimple**. The root Node is embedded inside the GameObject SofaContext and the two others objects are identified by the SofaNode: Cube and Cylinder.

Finally, the SofaDAGNode inspector allows to see the number of components handled and allows also to translate, rotate and scale the object in the **SOFA** world.



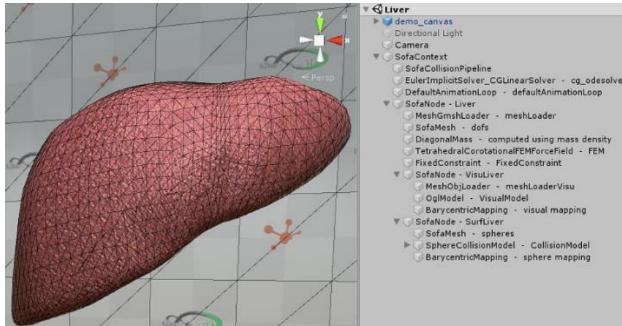
SOFA scene parsing

First approach to create a **SOFA** simulation scene inside **Unity3D** is to load an existing SOFA simulation scene. This is an XML file with .scn extension describing the graph architecture and the components.

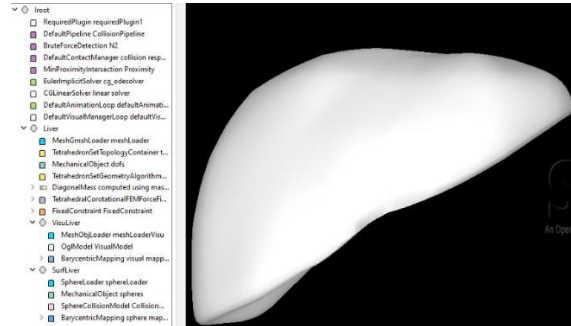
First thing to do is always to create the **SofaContext GameObject**. Then, use the **Load Scene button** to import a scn file. *Note that the gravity and timestep parameters will be overwritten by the values loaded from the scene file. But it is possible change those value again after the load.* Last thing important to notice is that Unity and Sofa have the X axis inverted. Thus to have the

same view of the scene from Sofa and Unity, Scale “-1 1 1” should be applied in the SofaContext.

Once loaded, every Node of the **SOFA** graph will be transposed as SofaDAGNode and the component into SofaUnity GameObjects.



Liver scene loaded inside Unity3D with the hierarchy display



Same Liver scene loaded inside runSofa GUI

3.2 - SOFA Object creation

Like in the **SOFA** simulation scene parsing. The first step to do is to create the **SofaContext GameObject**.

From that step, it is now possible to create several *SOFA Objects* through the UI Hierarchy panel. It can be seen as a higher API level which will create all the **SOFA** DAGNode and components to create the object.

The first type of objects are simple primitive like cubes, cylinders or planes. The second type are specific SOFA objects like collision objects.

This feature is still under development and might not work as expected.

In addition to the `SofaPrimitiveObject`, some specific objects will be provided. For the moment this section is very limited.

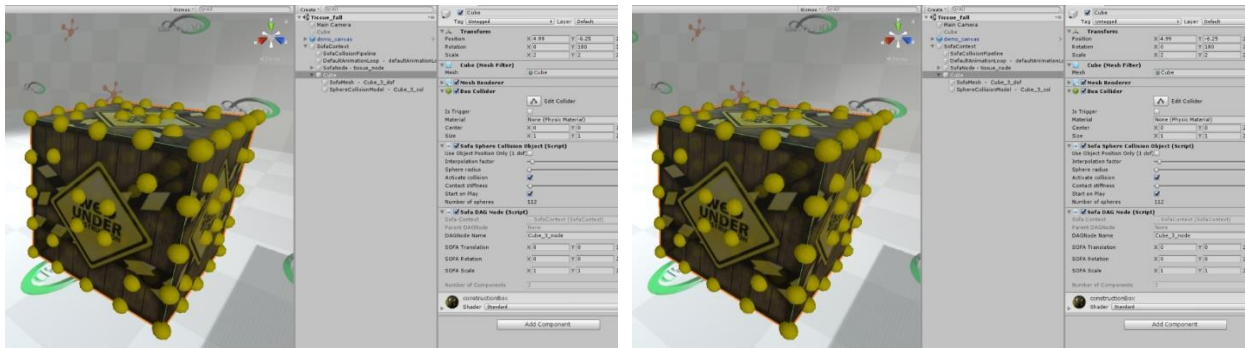
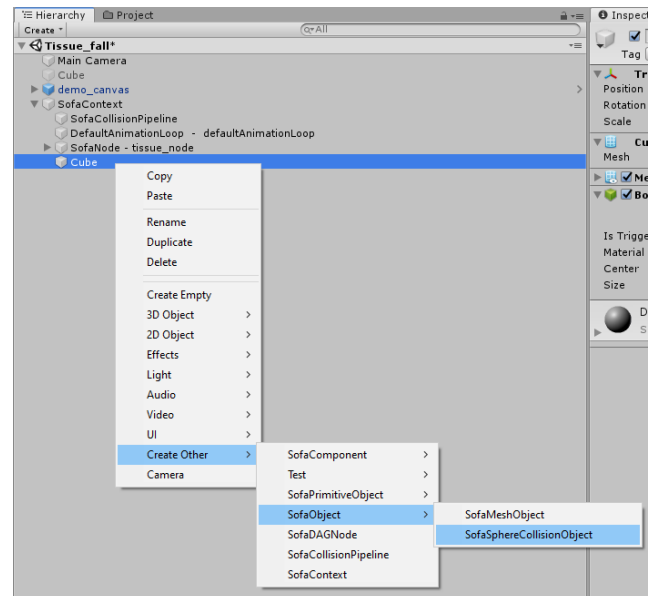
The first noticeable object is the possibility to create a sphere collision model on top of **Unity3D** objects.

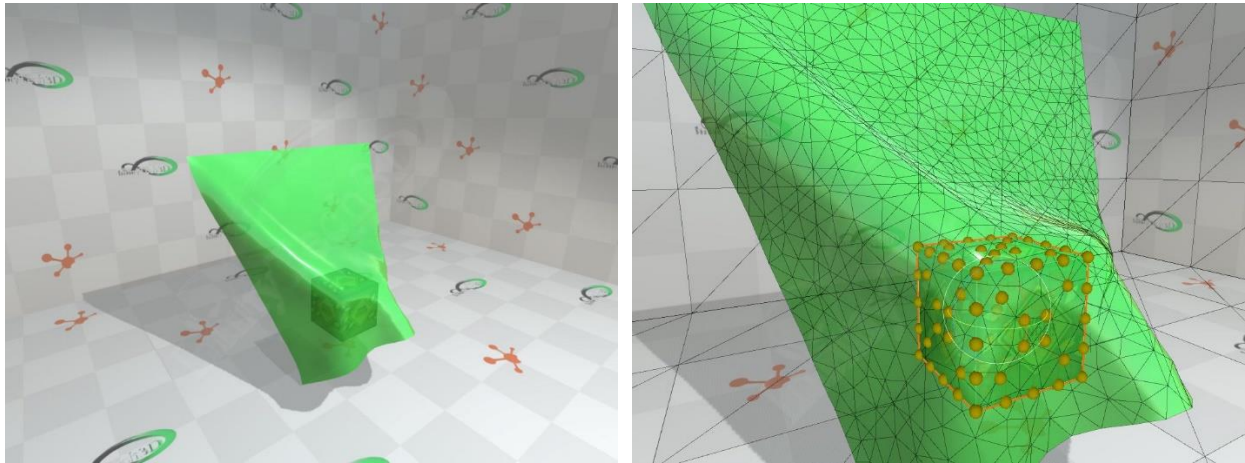
The **SofaSphereCollisionModel** will create spheres that can be seen as gizmos on the Unity side which are mapped to collision sphere inside the Sofa simulation scene.

The demo scene *Tissue_fall* is an example of that feature.

Several parameters can be set:

- **Interpolation factor:** a factor to compute the number of sphere to create on the given geometry.
- **Sphere radius:** radius of the collision spheres.
- **Activate collision:** collide or not with the other objects.
- **Contact stiffness:** the force of the repulsion used by the collision sphere.





3.3 – SOFA Graph Edition

This API is still work in progress. It correspond to the deeper level of API where all edition between SofaUnity components will be possible. It can be seen as an alternative to the scene file or the python scripts usually used by SOFA users.

For the moment those features should only be used to finalise the simulation graph of a loaded scene. Several components, described below, are already available and it is also possible to manually add new SofaDAGNode in the simulation graph.

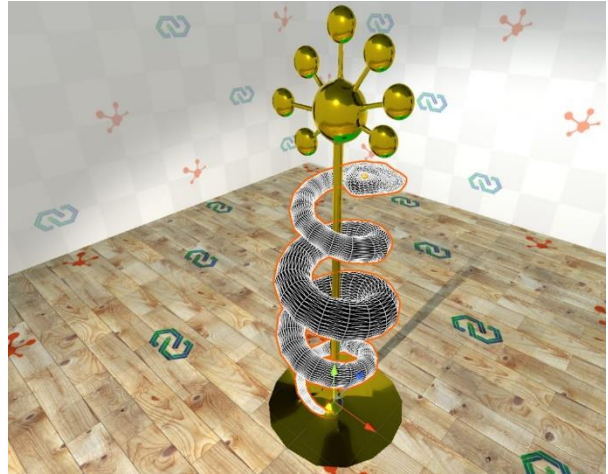
8. Available Features

The **SofaUnity** Assets can be used to perform simple physical simulation of deformable 3D models in Unity to more advanced mechanical behavior. Below is a summary of all examples and demo scene available in this package. Full information can be found online at the different links.

Physics – Basic Examples

This section describes the scenes located in *SofaUnity/Scenes/Examples/*. These scenes illustrate basic physics that can be simulated using the SOFA framework in Unity. Most of them are examples present as is in the SOFA repository.

[Full documentation online](#)



Physics – Simulation interactions

This section describes the scenes located in *SofaUnity/Scenes/Demos/Interaction*. These scenes illustrate more advanced physics simulation scene with Unity interaction either using the mouse, the keyboard or user interface. The demos described below are:

1. **Demo_01_LiverInteraction**
2. **Demo_02_LiverInteraction-HD**
3. **Demo_03_CubeCarving**
4. **Demo_04_CubeAdvancedCarving**
5. **Demo_05_LiverCut**
6. **Demo_06_LiverCut-CUDA**

[Full documentation online](#)

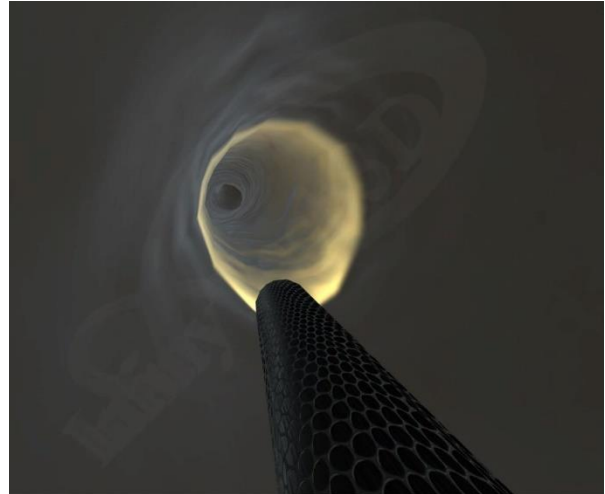


Endoscopy - SOFA Beam Adapter plugin

This section illustrate the scenes located in */Scenes/Demos/Endoscopy/BeamAdapter/*

Those scene integrate the SOFA BeamAdapter plugin allowing to have a dynamic implementation of the FEM beam elements. More details can be found on [github](#).

In those examples, Unity is used to control the BeamAdapter navigation and also attach camera at the tip of the tool.



[Full documentation online](#)

Endoscopy – Virtual capsule navigation

To use the capsule navigation system, a single rigid degree of freedom is simulated on the SOFA side. This is what is done in the “*SofaNode – Capsule*” Node. The SofaMesh is composed of a single position without mesh and a sphere collision.

On the Unity side, a 3D object is used such as a capsule, but any 3D mesh could be used. The *SofaCapsuleController* script is used on that object to link the position of the SOFA rigid object with this Unity GameObject Transform.



[Full documentation online](#)

9. Additional Assets

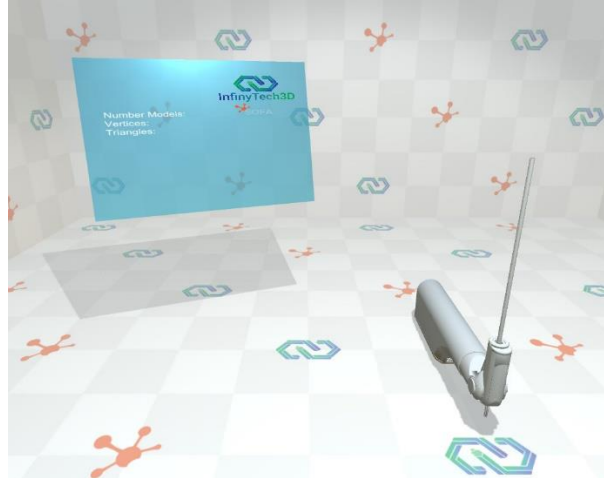
This section gives a quick overview of additional assets adding specific features to the SOFA Unity integration. Those assets can be purchased on the Unity Assets Store or requested from [this online form](#).

Those assets have different licenses and price which can be found on this [license page](#).

Haptic device simulations



[This asset allows to use SOFA Geomagic plugin inside Unity3D](#)

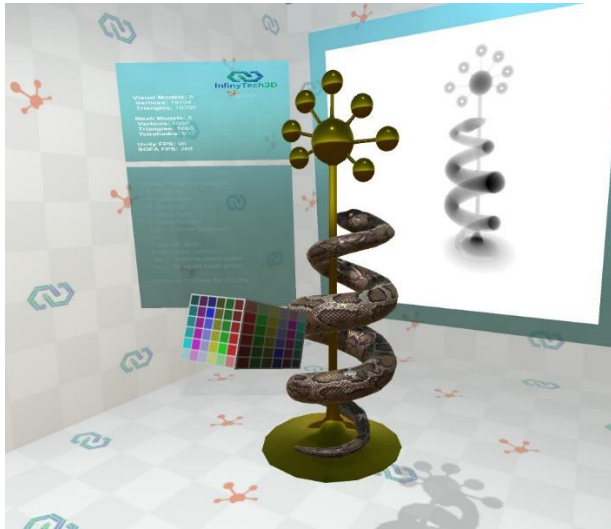


[This asset integrates the SOFA Follou Haptic Avatar plugin](#)



[This asset integrates the SOFA Haply Robotics Inverse3 plugin](#)

Medical Imaging rendering



[This asset allows to compute the fluoroscopic images of 3D models inside SOFA in real time while the simulation is running](#)



[This asset allows to compute UltraSound images of 3D models inside SOFA in real time while the simulation is running](#)