Integrating Gaussian Splatting in SOFA

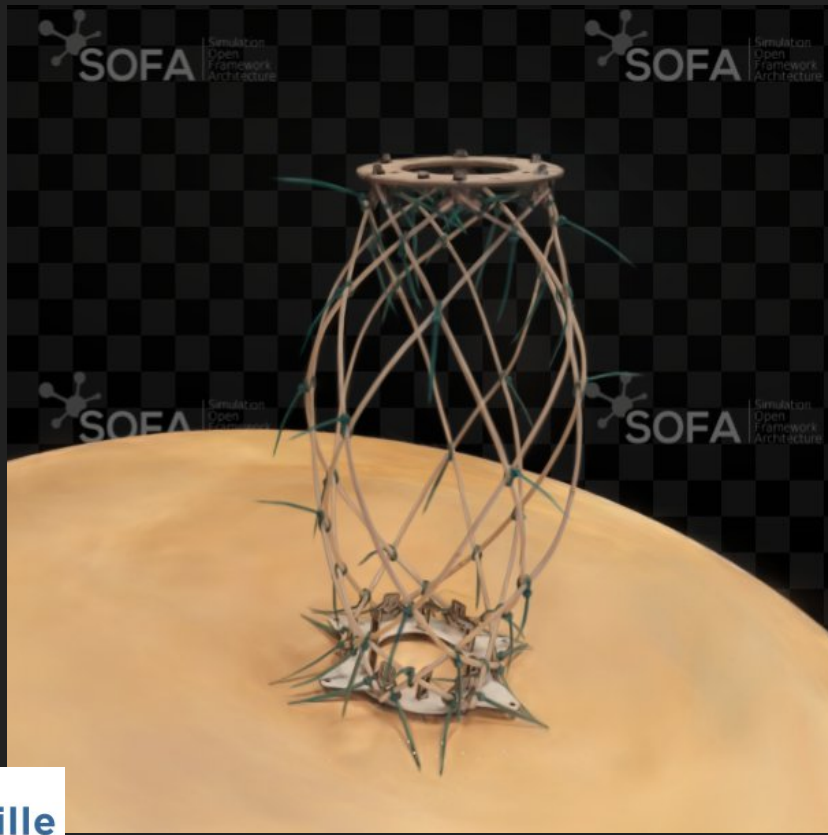
A join effort from the CRIStAL engineering teams on VR, Robotics and Simulation.

Damien Marchal
DEFROST/PADR

Bryan Emery
PIRVI/PADR

Maxime Duquesne
PRETIL/PADR

Mine2Rien RetEx talk 06/05/2026



The story behind GS at CRIStAL

- Show we can have high-tech project (robotics, RV) with more “positive” environmental impact ... RV

- Evaluate the environmental cost of “doing” our research (building, mobility, technical plateforms)

Output:

- train environmental robots (drones to gather biodiversity data)
- study psycho-socio sensibility to environmental degradation through sensible experience in RV
- study psycho-socio adhesion to urban renovation relatively to how sensible the presentation is (paper, blueprint, image, RV)
- quantify realism in a sensible experience.

All that rely on the capacity to generate large 3D content (sound and image).. Generative IA for RV

-> these world need to be “realistic” and dynamic/interactive, so how to get there ?

Gaussian Splatting

https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

Gaussian Splatting

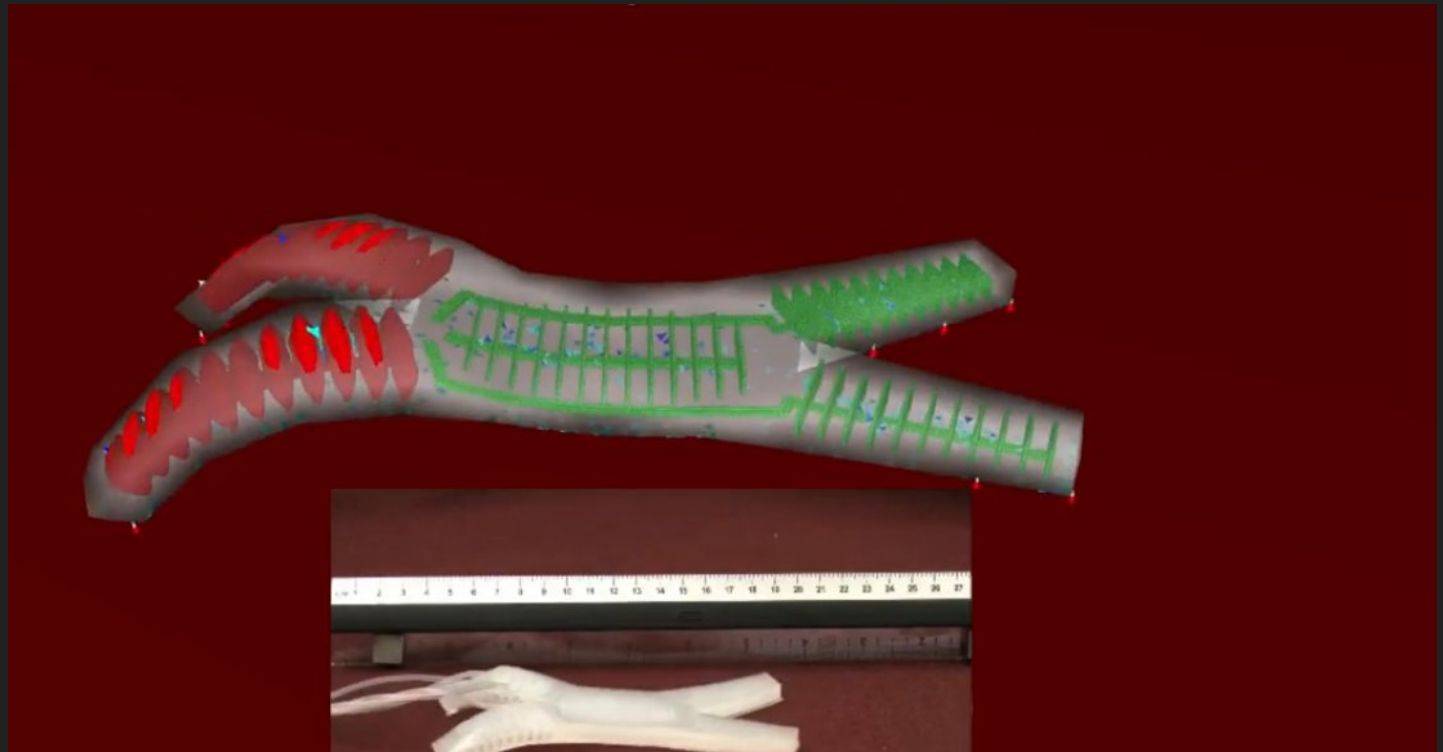
https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

https://www.realtimerendering.com/kesen/sig2025.html

https://www.youtube.com/results?search_query=Gaussian+Splatting

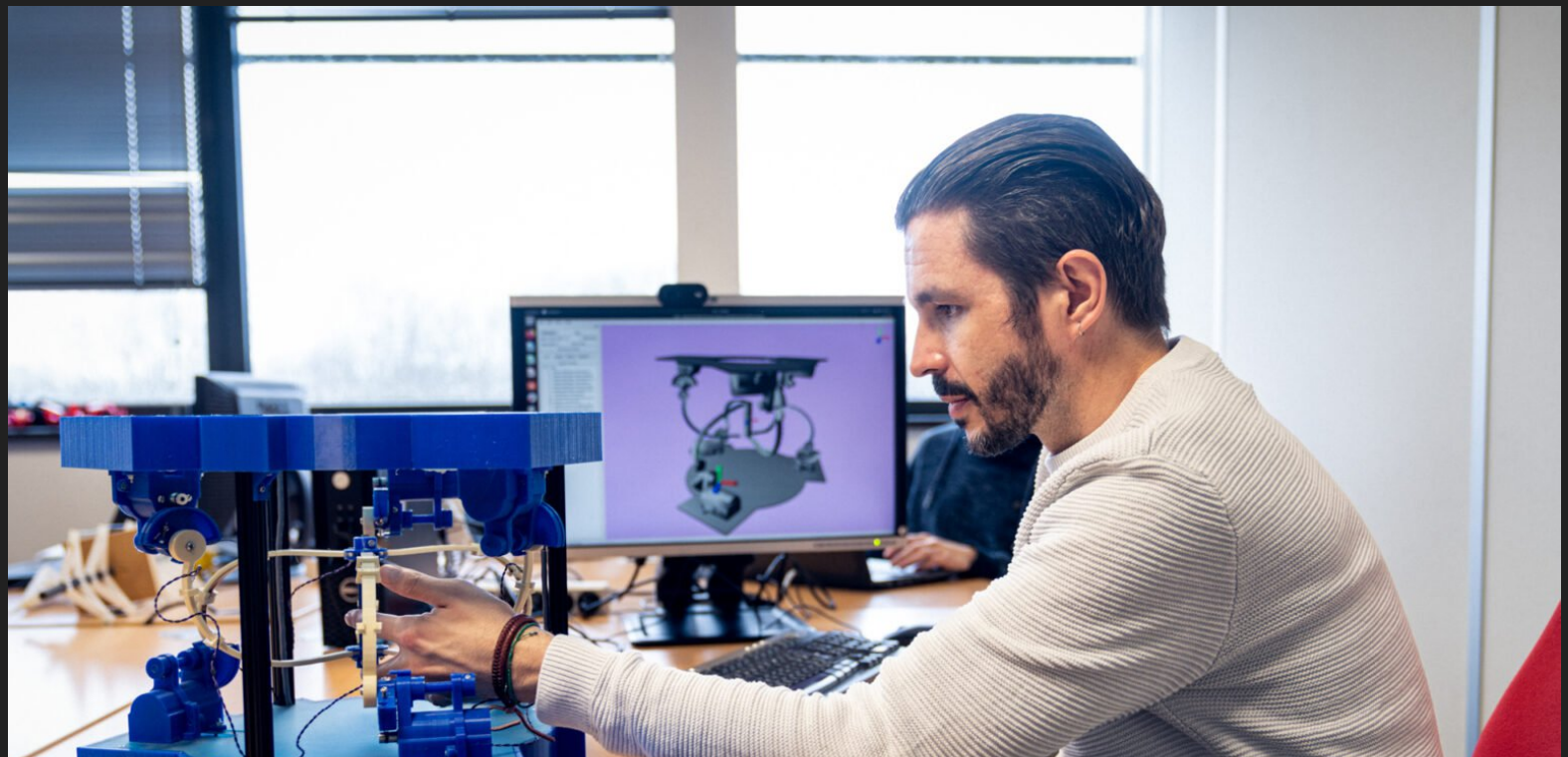
Introduction

DEFROST: DEFormable RObotics SofTware



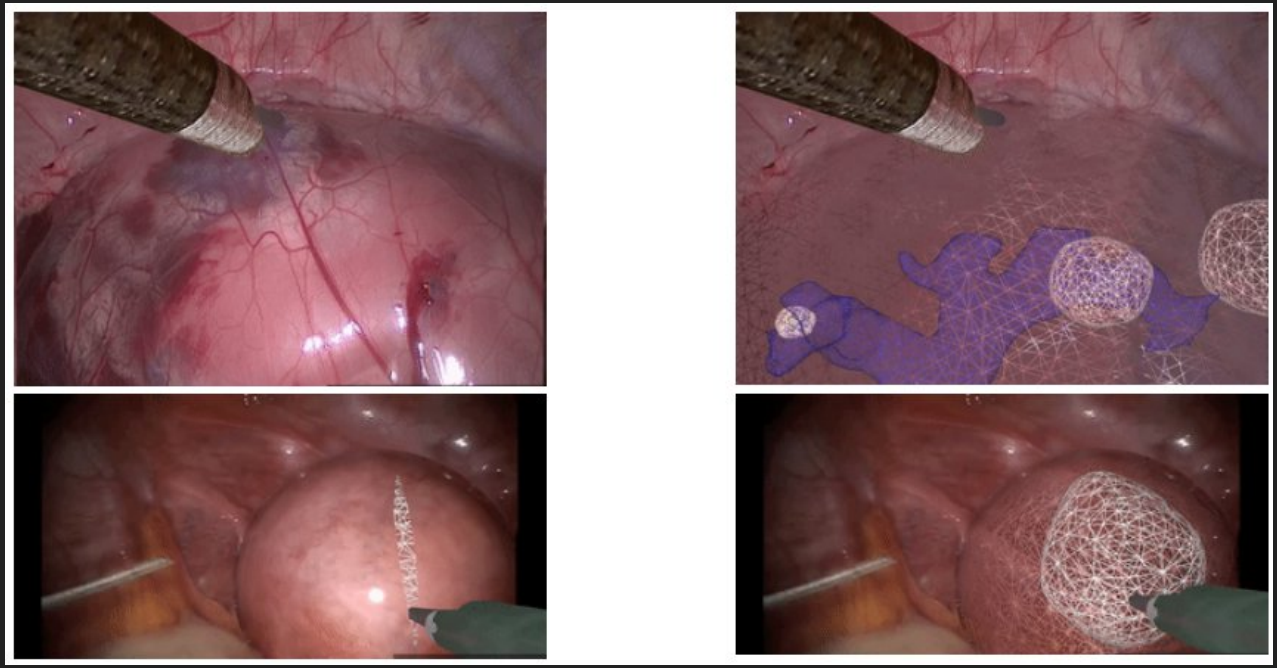
Introduction

DEFROST: DEFormable RObotics SofTware

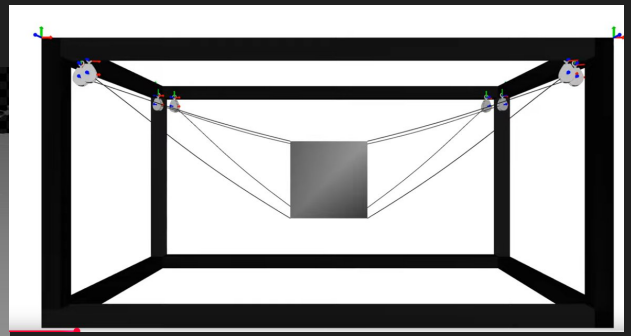


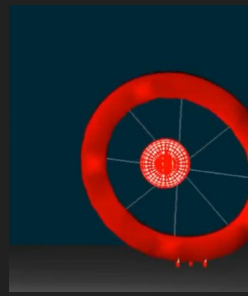
Introduction

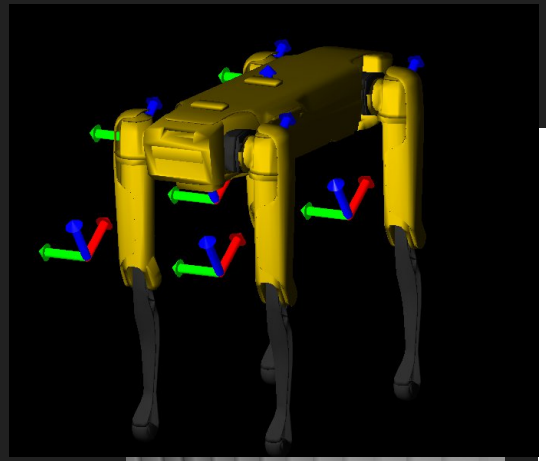
SOFA Framework: https://www.sofa-framework.org/
A multi-physics simulation core initially designed for bio-medical/material

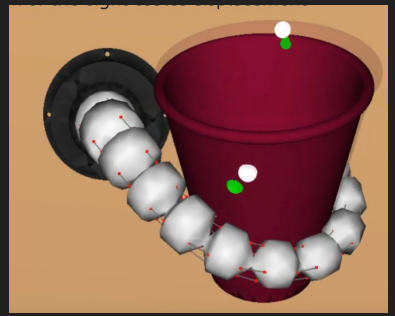


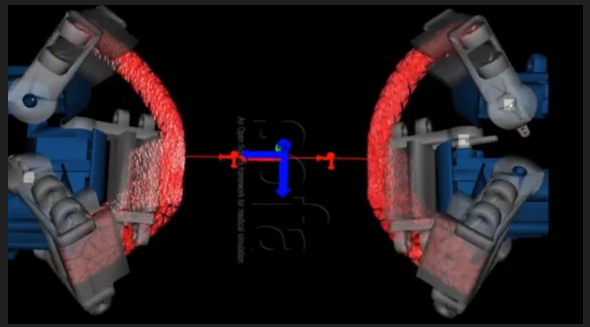
The rendering of our simulation looks like in the 90’

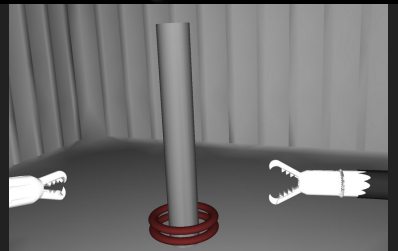




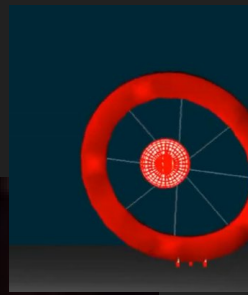


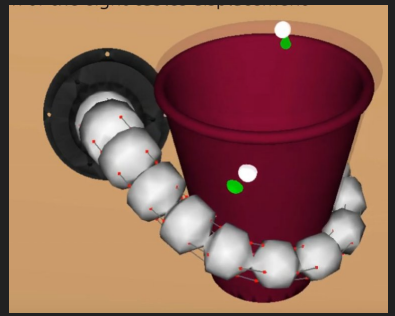






The rendering of our simulation looks like in the 90’





So we feel like in Jurassic Park

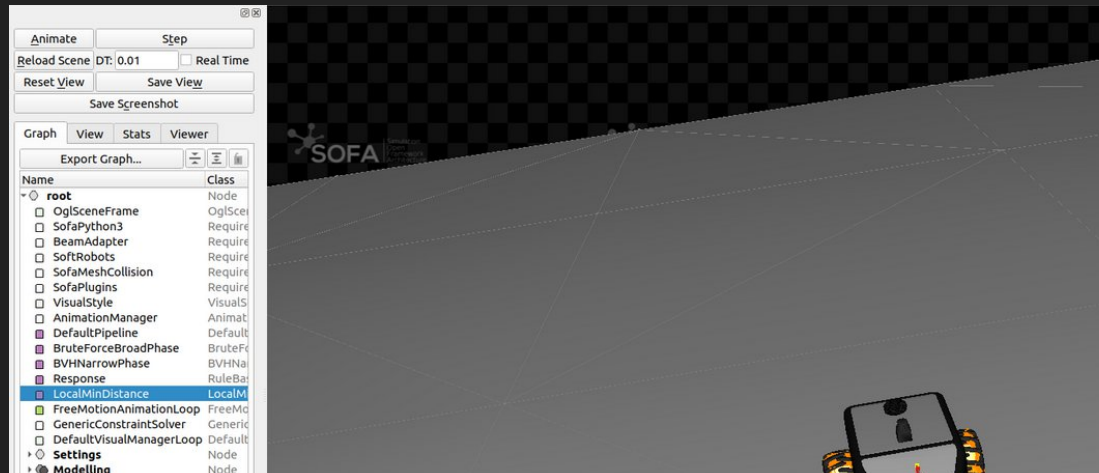
Two reason of our “poor quality” scene rendering

As it is not crucial for the work we are doing it is not prioritary to invest time

So:
- the default SOFA Renderer is old fashionned
- high end 3d rendering imply high end modeling which very costly
(unless you have free generic assets)

More and more we use simulation
to generate training datas to build
ML based controller

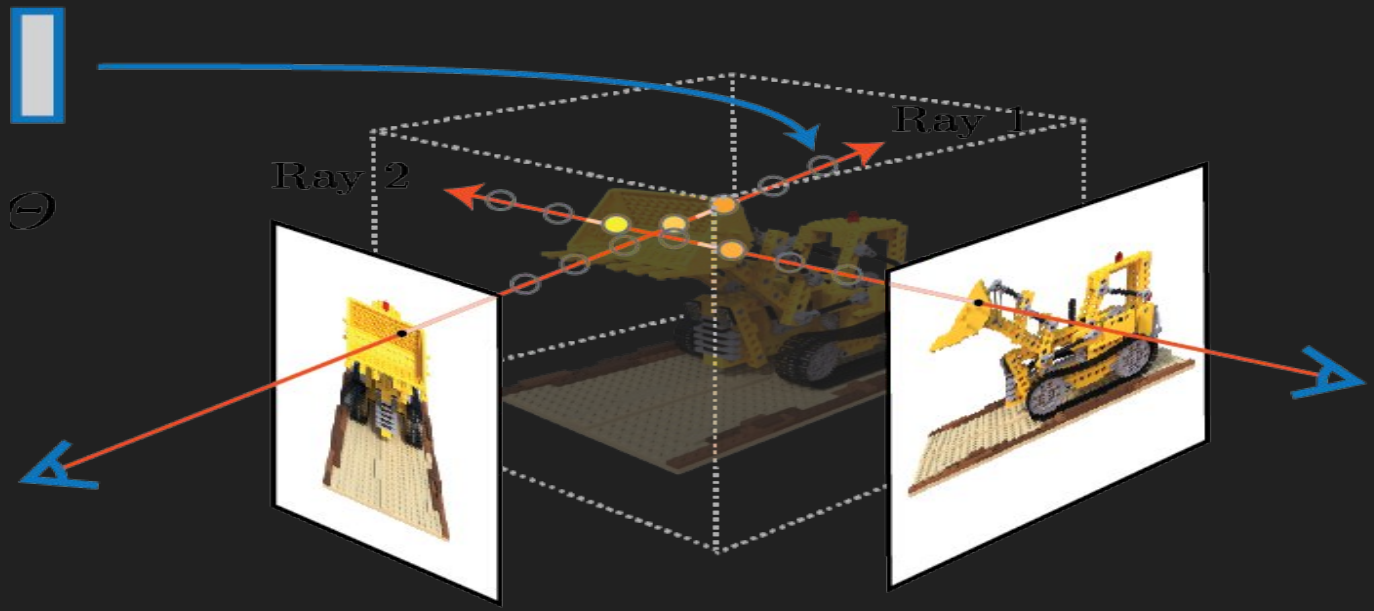
So we need realistic synthetic dataset
Including the robot “environment”



Reconstruction based on geometry: 3D Scanners

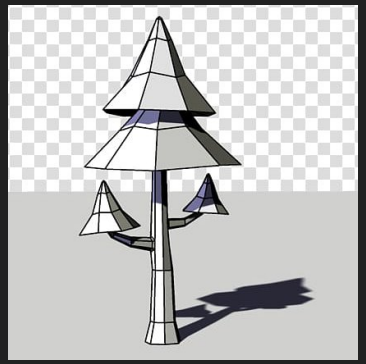
Reconstruction from images

Reconstruction from images

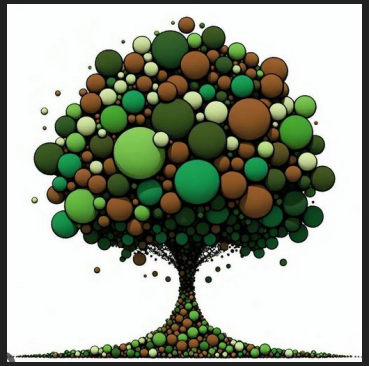


Encoding geometry

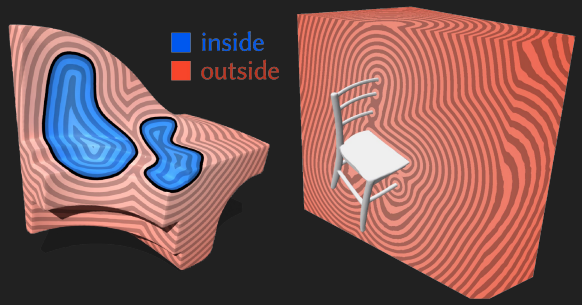
Meshes



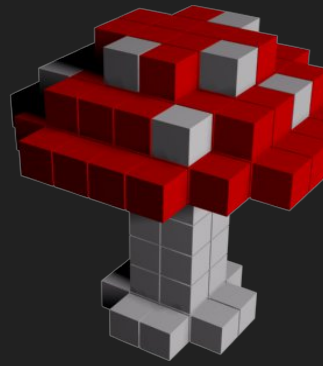
Gaussian Splat



Implicit Function
(Signed Distance Field)



Voxel

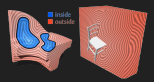


Implicit Function
Neural SDF,
Neural Radiance FIeld

Reconstruction based on inverse rendering

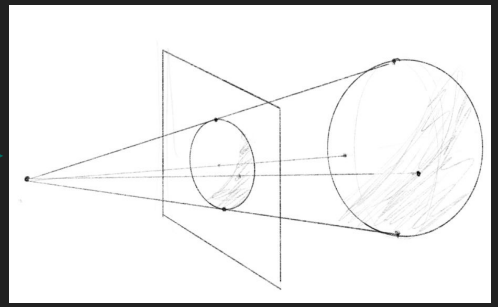
Model differentiable

GaussianSplat



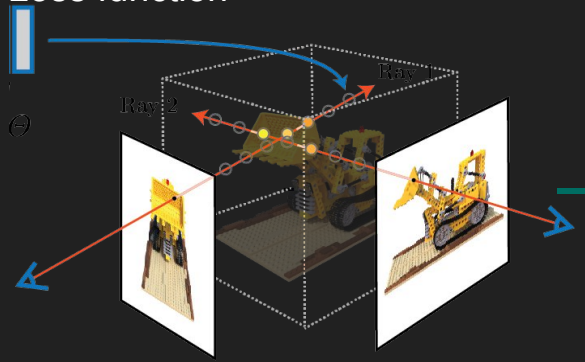
Nerf & NeuralSDF

Renderer differentiable



Measure the difference between rendering & capture

Loss function



Iteratively minimize the loss function

(Need the computation of the derivatives of the Loss function against model “parameters”, how a small change on model impact the pixel of the image)

Reconstruction based on inverse rendering

Video Tutorial made by Bryan Emery
https://www.youtube.com/watch?v=9zGEv0dGvOw&t=200s

Step 1: record a video

Step 2: use COLMAP to locate camera position

Step 3: use *nerfstudio* or *brush* to generate a Gaussian Model

Step 4: use blender to segment the data set

Step 5: use in your preferred software

Reconstruction based on inverse rendering

Video Tutorial made by Bryan Emery
https://www.youtube.com/watch?v=9zGEv0dGvOw&t=200s

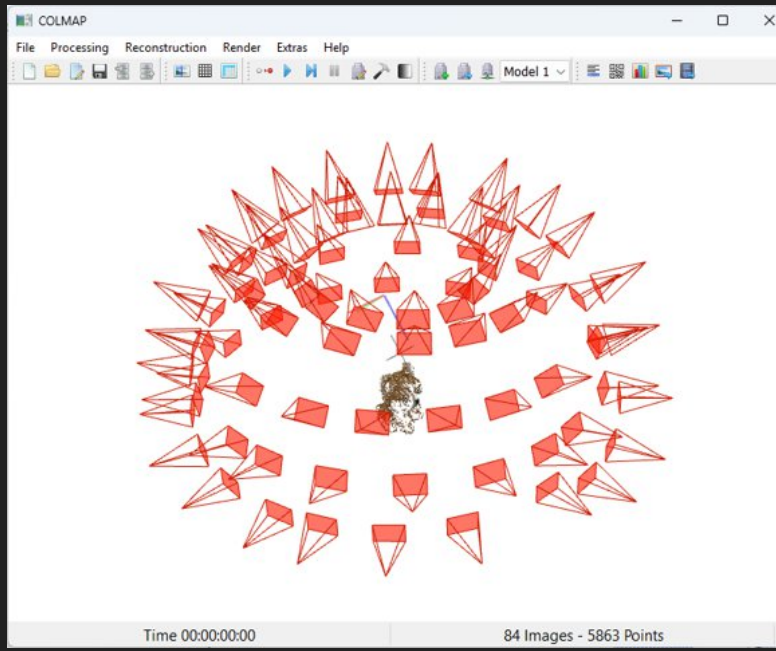
Step 1: record a video

Step 2: use COLMAP to locate camera position

Step 3: use nerfstudio or brush to generate a Gaussian Model

Step 4: use blender to segment the data set

Step 5: use in your preferred software



Reconstruction based on inverse rendering

Video Tutorial made by Bryan Emery
https://www.youtube.com/watch?v=9zGEv0dGvOw&t=200s

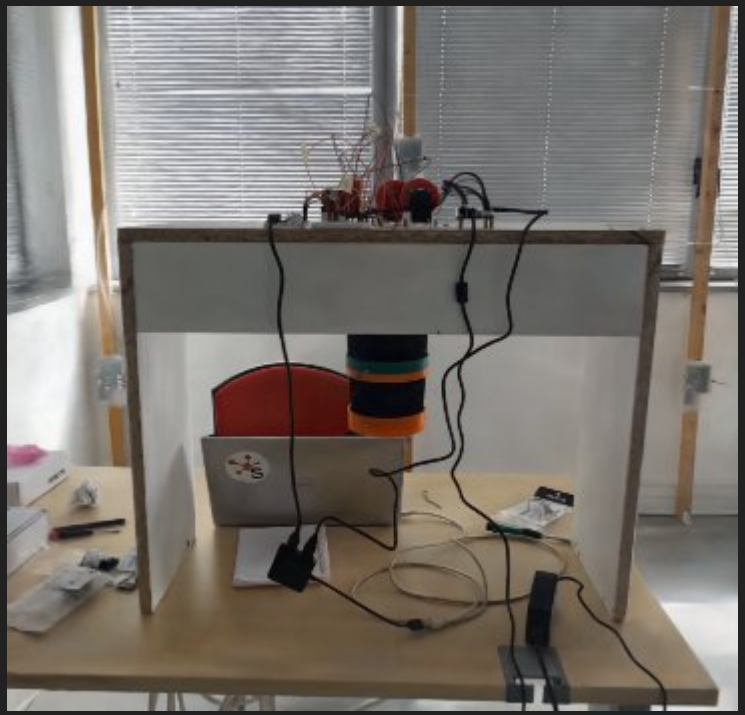
Step 1: record a video

Step 2: use COLMAP to locate camera position

Step 3: use nerfstudio or brush to generate a Gaussian Model

Step 4: use blender to segment the data set

Step 5: use in your preferred software



https://www.kiriengine.app/share/3dgs?taskId=2039344906559815680

Sofa.PointCloud 1.0

github.com/CRIStAL-PADR/Sofa.PointCloud/

This plugin implements Gaussian Splatting rendering in SOFA to simplify the use of 3D model or 3D environment generated from videos and reduce sim2real gap.

1 - Make a video

2 - Generate a gaussian model using tool like

gsplat

nerfstudio

KIRI Engine

3- Import the model in your SOFA scene, connect it to your physic and **simulate !**

Sofa.PointCloud v 1.0

The plugin currently features the components:

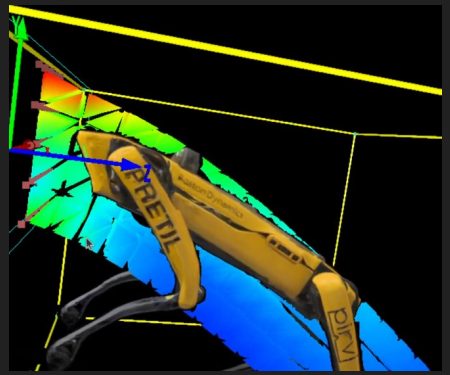
- PointCloudContainer: stores the geometrical description of the point cloud
- ~~PointCloudTransform: apply rigid transformation to a pc container~~
- PointCloudVisualModel: make a container visible and apply transform to all or subset of splats.
- PointCloudRenderer: renders all the PointCloudVisualModels in the 3D view

Minimalistic example:

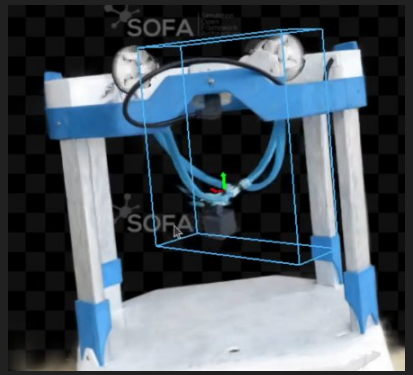
```
<?xml version="1.0"?>
<Node>
    <RequiredPlugin name="Sofa.PointCloud"/>
    <InteractiveCamera name="camera"/>
    <PointCloudRenderer camera="@camera"/>
    <PointCloudContainer name="container1" filename="splats/spot.ply"/>
    <PointCloudVisualModel name="visual" geometry="@container1"/>
</Node>
```

Sofa.PointCloud v 1.0

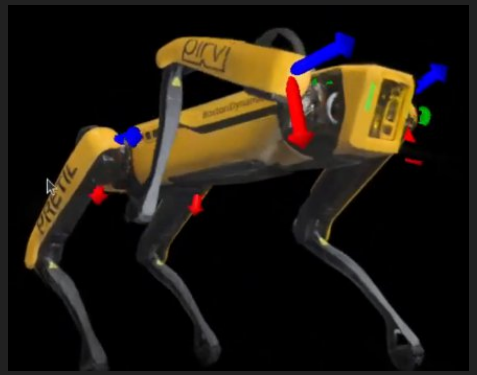
GS model can be deformed using an input data field with rigid frames:
can be driven from: *MechanicalObject<Rigid>, beams, cosserats, shells, rigid transform and volumetric deformation by using mapping like BarycentricMapping<Vec3, Rigid>*



TetrahedronFEMForceField
BarycentricMapping<Vec3,Rigid>



Legs: Beams<Rigid>
Cube: MechanicalObject<Rigid>

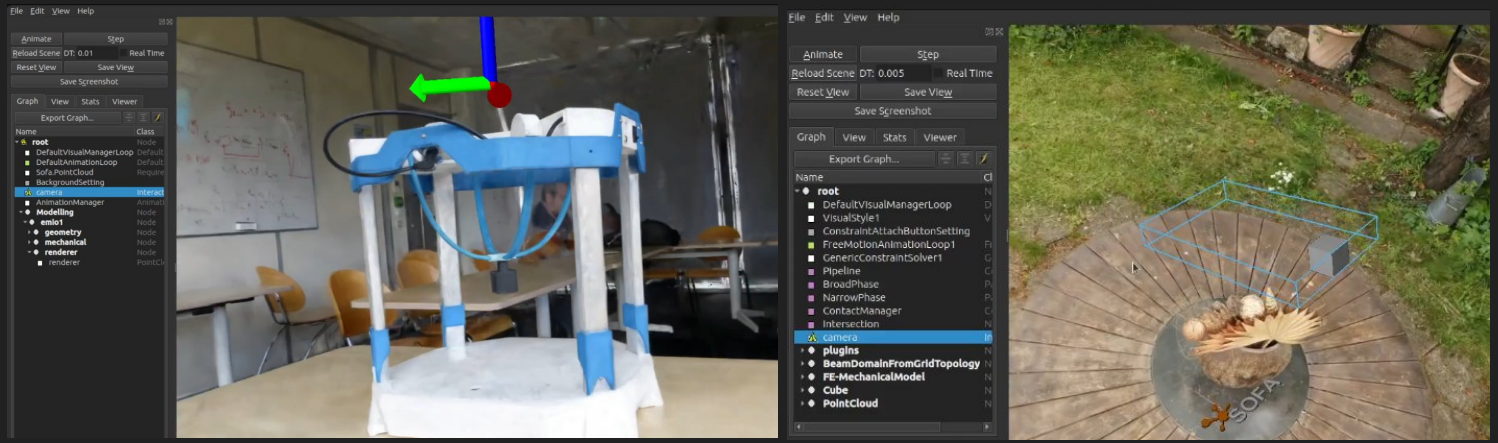


Simulated model loaded from URDF
Using the Sofa.RigidBodyDynamics

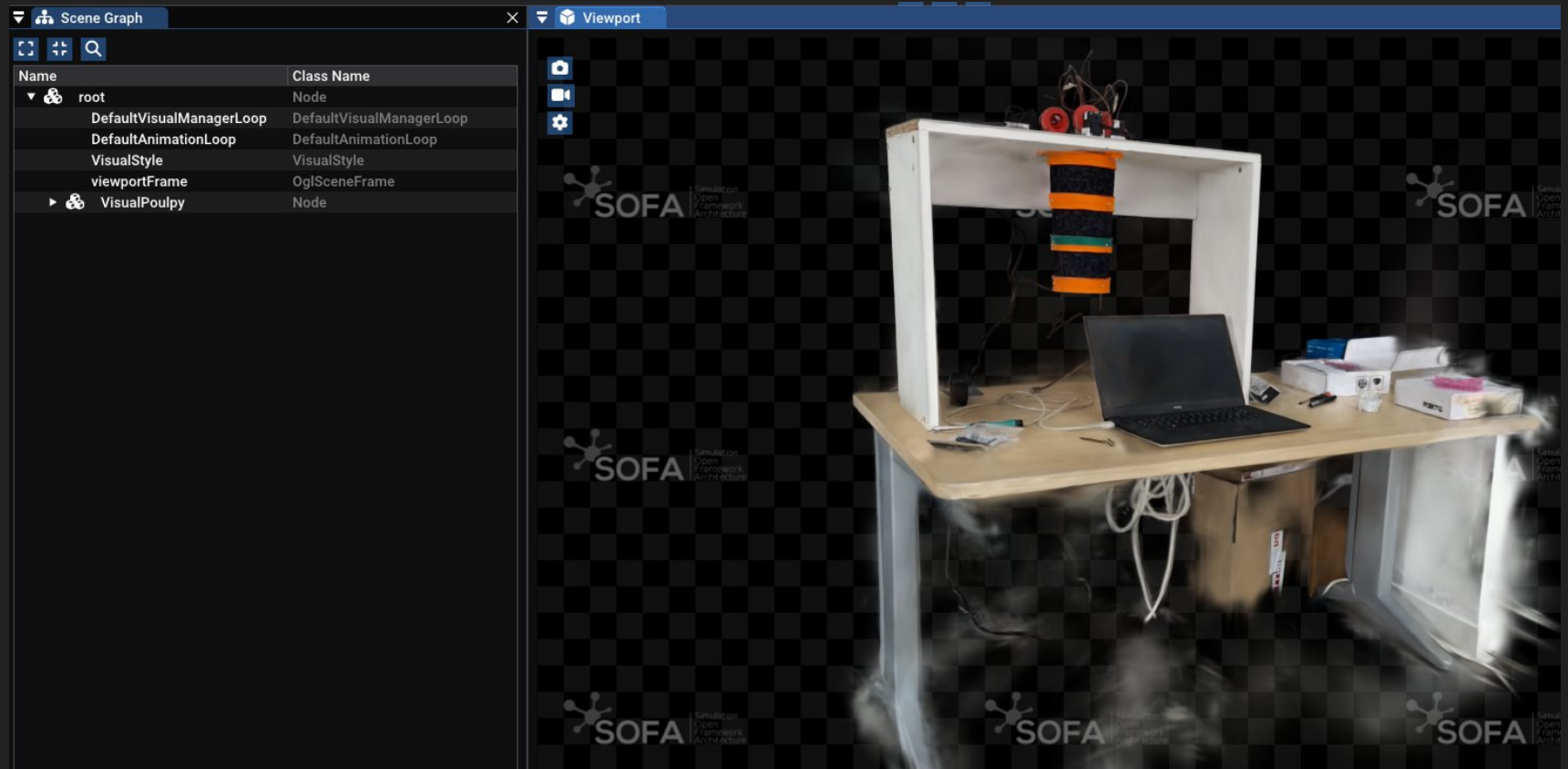
https://www.youtube.com/watch?v=ozRv9O6N_vs&list=PLYhlzFXHKQNM6hGw3cx_sIgVKAdVQFc4D

Sofa.PointCloud: v 1.0

A CUDA based renderer so we can render full environment in realtime [DONE]

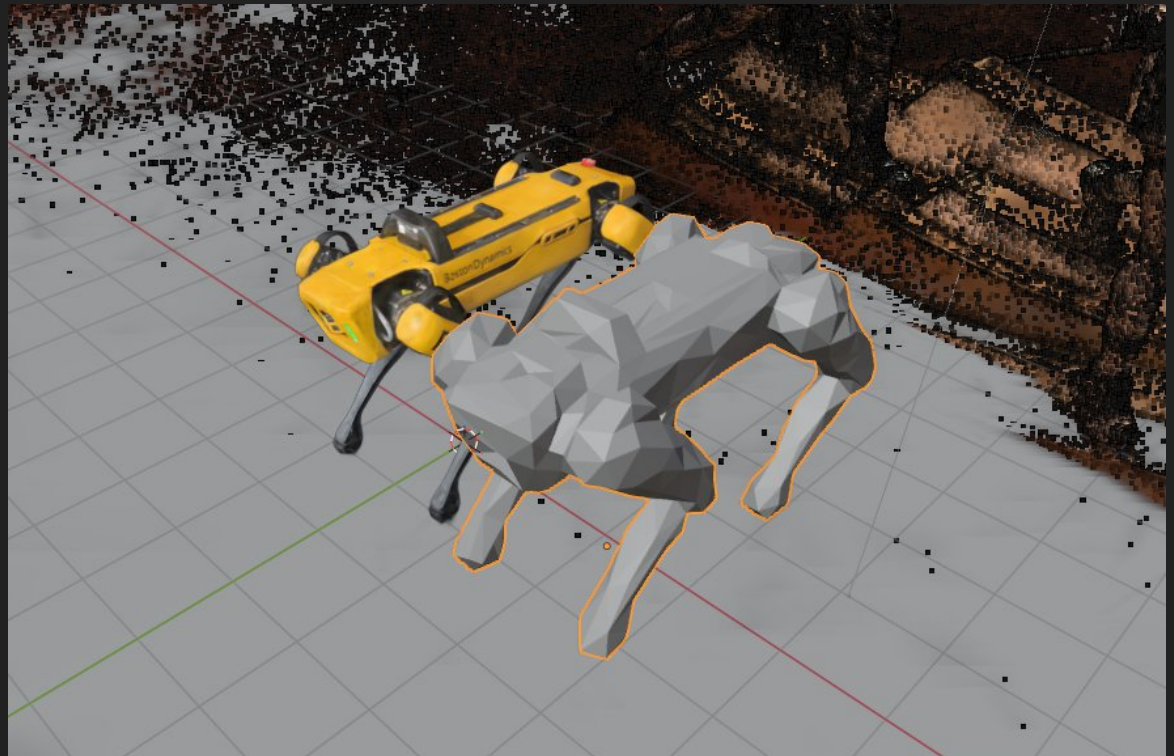


Sofa.PointCloud: v 1.0



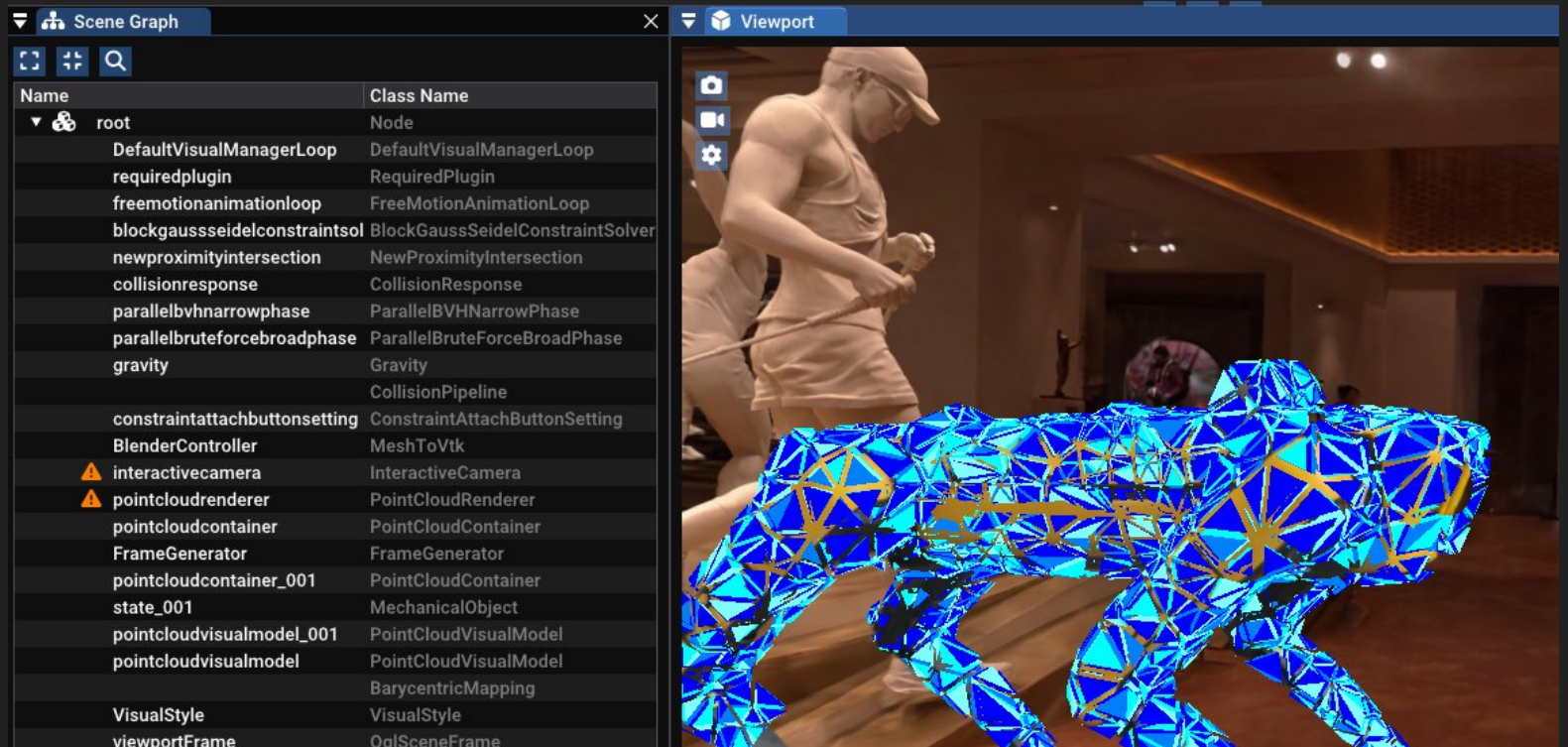
Sofa.PointCloud: vWIP (WIP)b 2.0

Collision detection on GS, NERF and NeuralSDF => more meshes



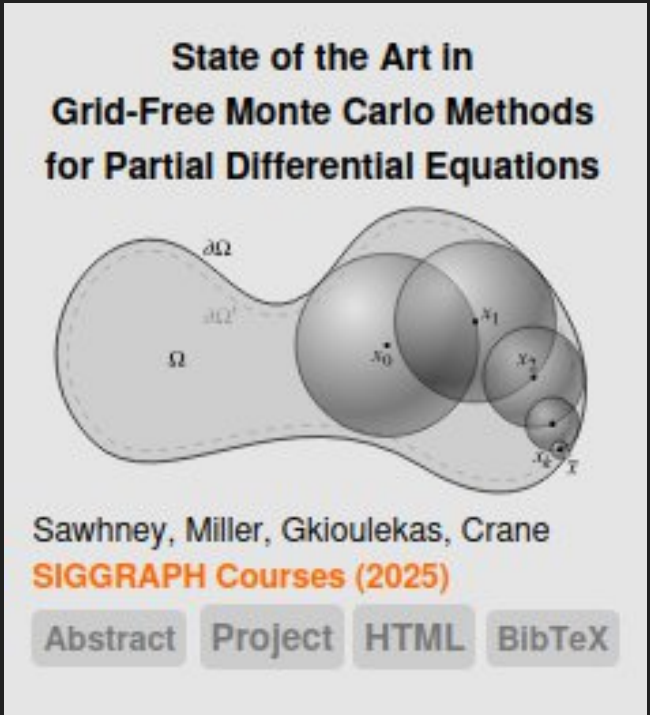
Sofa.PointCloud: v 2.0 (WIP)

Physical simulation on GS, NERFS => no more mesh



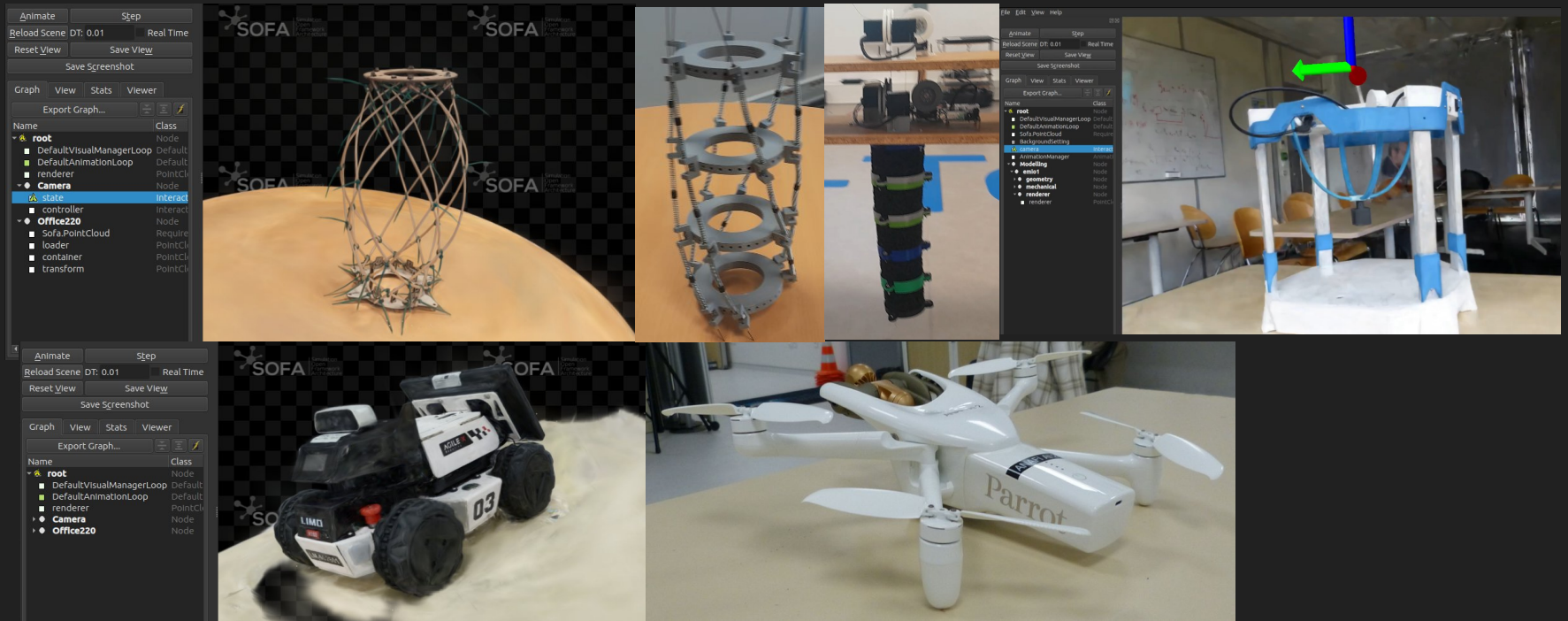
Sofa.PointCloud: v ?.0 (WIP)

Gridfree solver for PDE on NERFS structures ?



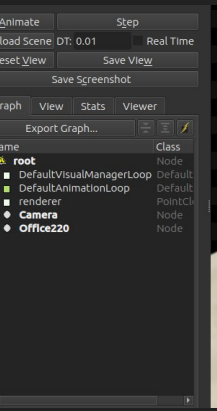
Sofa.PointCloud: v 2.0

Set of models with SOFA prefab simulations



Sofa.PointCloud: v 2.0

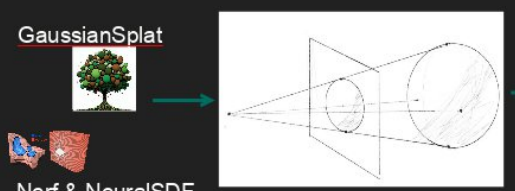
Set of models with SOFA prefab simulations

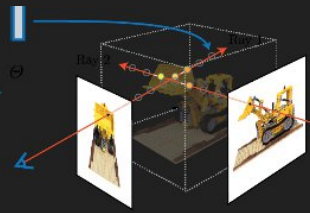


Sofa.PointCloud: v 2.0 (WIP)

Nicolas Senecaux: Automatic segmentation and up-sampling

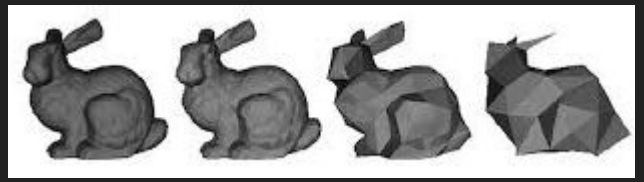
Biasing





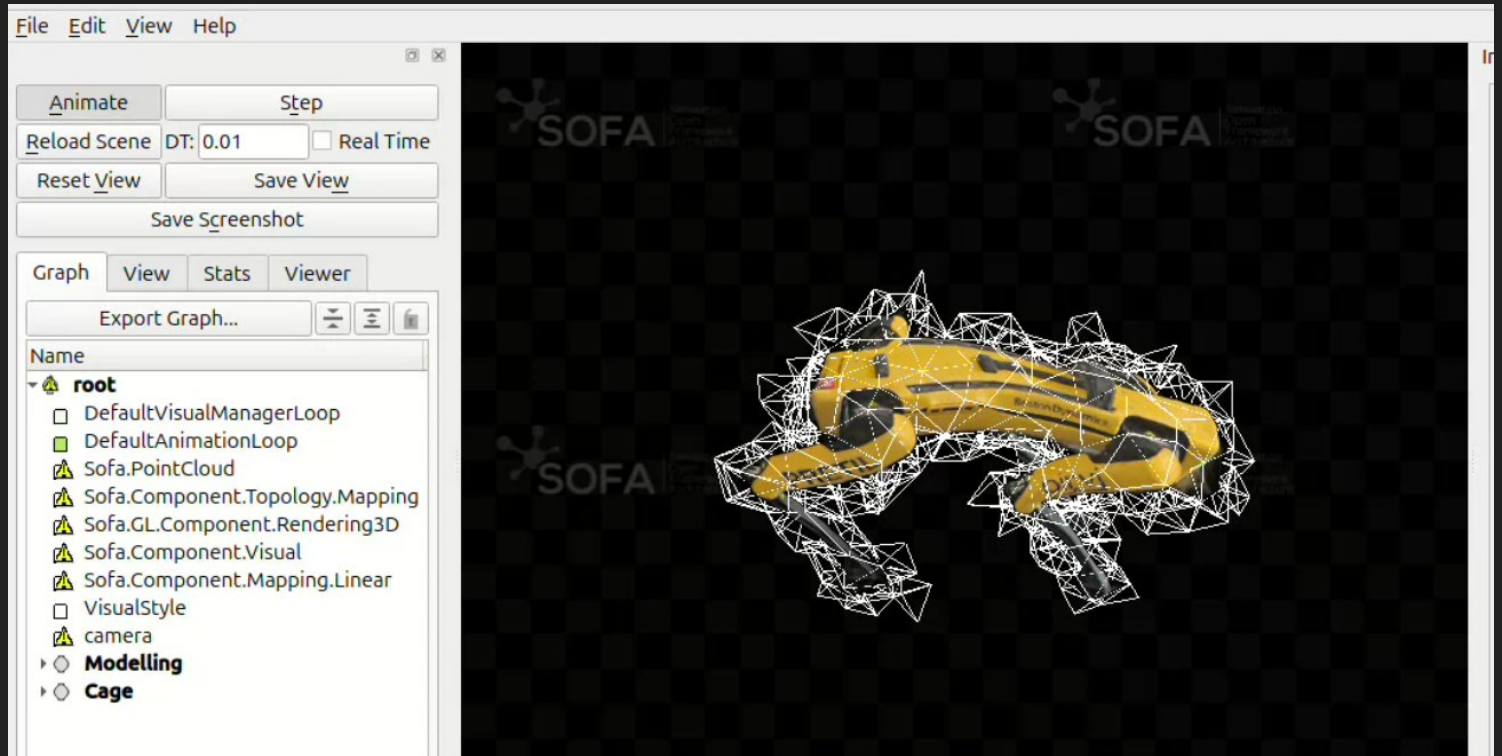
Sofa.PointCloud: v 2.0 (WIP)

Nicolas Senecaux: Level of Detail



Sofa.PointCloud: v 2.0 (WIP)

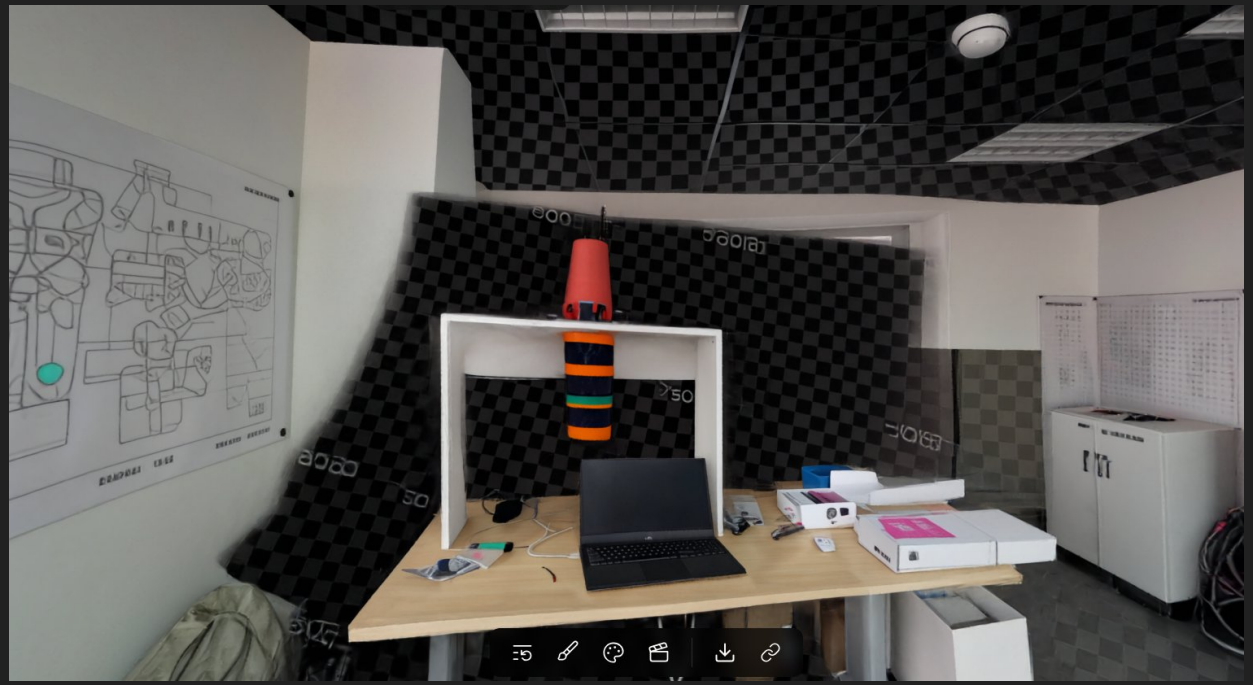
Mateo Charles-Mennier: Deformation & Collision



Sofa.PointCloud: v 2.0 (WIP)

Valentino Pagani: Automatic segmentation and up-sampling

Generative IA for envrionnement modelling



https://marble.worldlabs.ai/world/00b56a2e-60a2-42df-a9f1-5b4316ded436

Sofa.PointCloud: Join us, help needed !

github.com/CRIStAL-PADR/Sofa.PointCloud/

damien.marchal@univ-lille.fr

Other things that would be really cool:

- relighting & shadow
- super-sampling
- filling-in the part the are missing in the video with generative IA
- use generative IA to produce our input videos from semi-prompt
- 4D reconstruction .....

Credits

The Sofa.PointCloud plugin has been developed within a join effort from the CRIStAL engineering teams on VR, Robotics and Simulation.

Sofa.PointCloud	3D GS generation	Video footage
Damien Marchal	Bryan Emery	Maxime Duquesne

Robots kindly provided by: PRETIL, PIRVI, DEFROST team and ComplianceRobotics

Emio and Spot SOFA scenes done by Eulalie Coevoet (ComplianceRobotics)

For third party libraries we rely on:
openGL GS renderer: https://github.com/panxkun/liteviz-gs
CUDA GS renderer: https://github.com/graphdeco-inria/diff-gaussian-rasterization