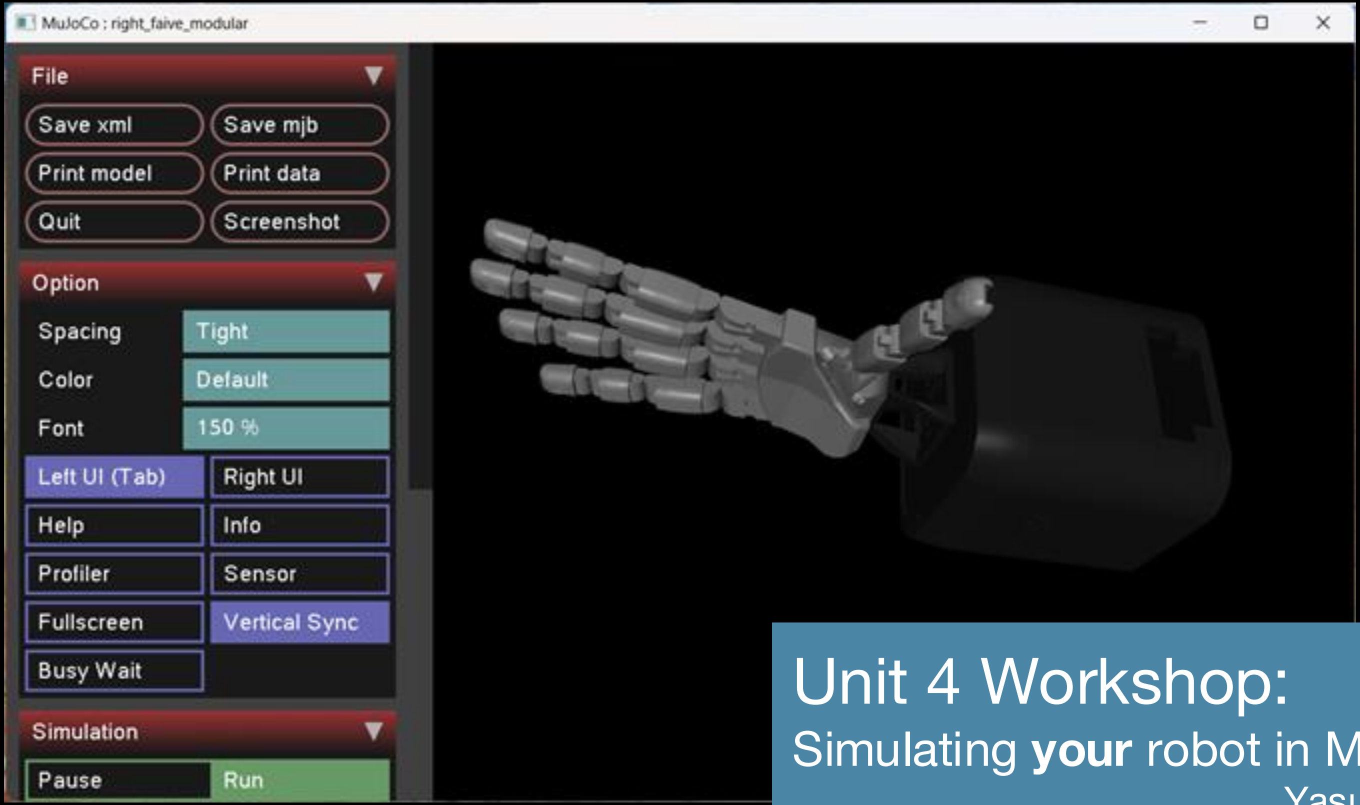
ETH zürich



Spacing Tight
Color Default
Font 150 %

Unit 4 Workshop:

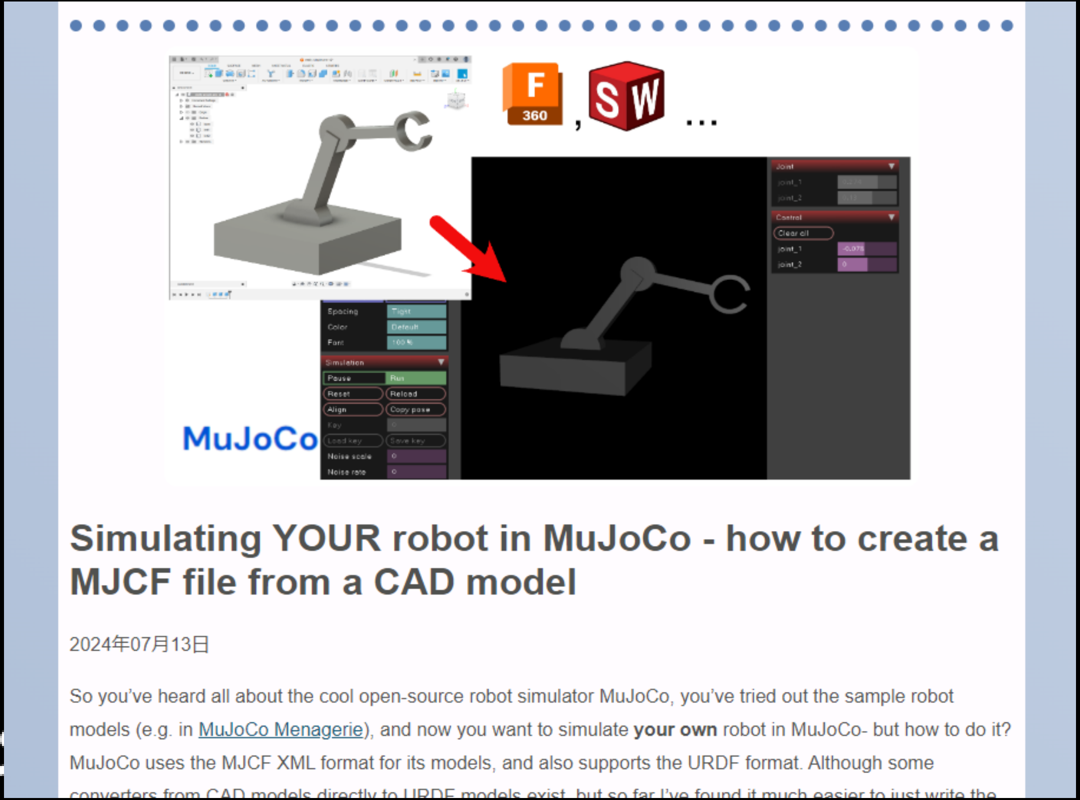
Simulating **your** robot in MuJoCo

Yasunori Toshimitsu

SoftRobotics Laboratory

For a more detailed blog post version of this tutorial...

https://yasunori.xyz/en/2024/07/13/mujoco-model-yourself.html

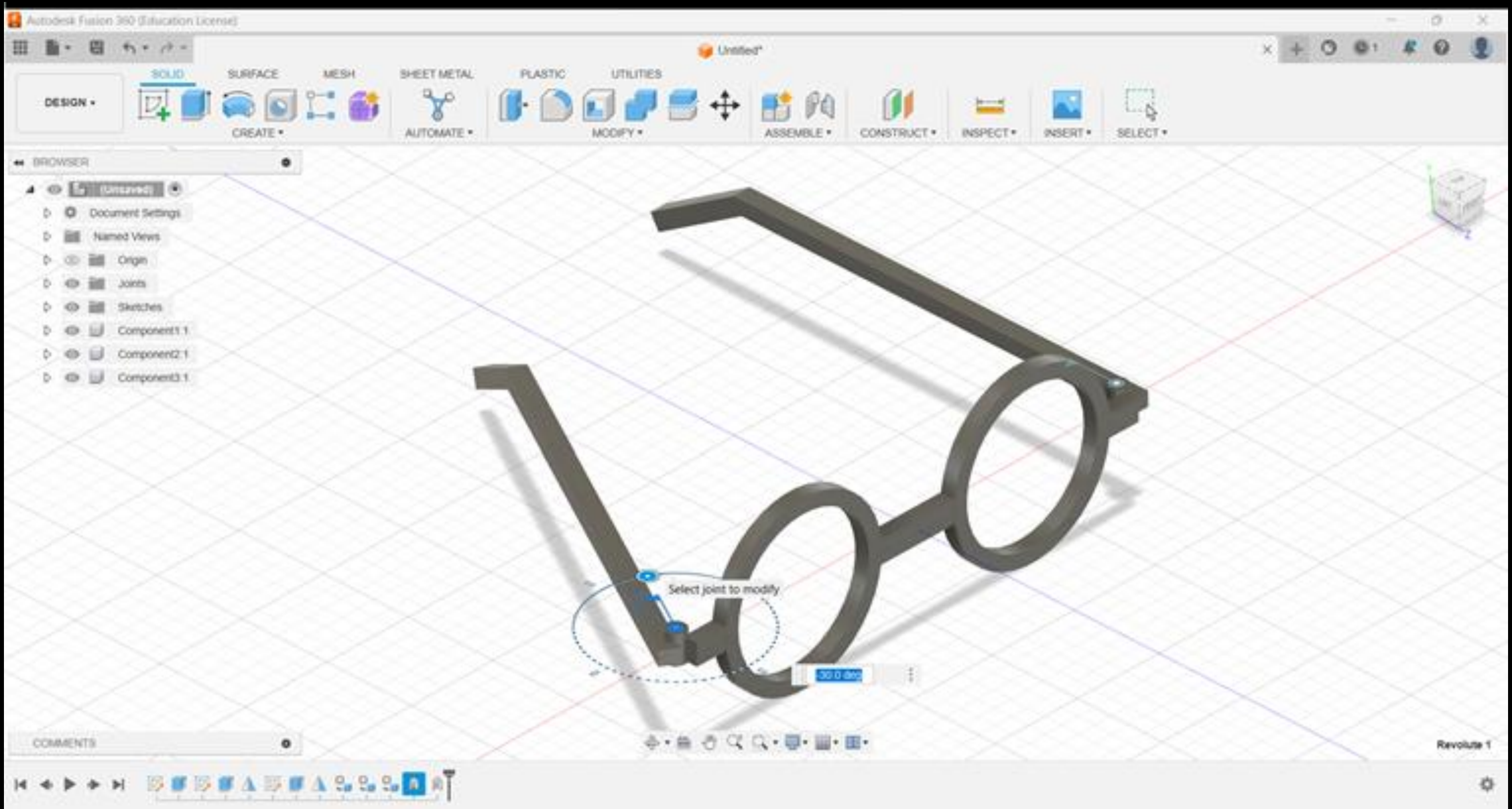


Simulating YOUR robot in MuJoCo - how to create a MJCF file from a CAD model

2024年07月13日

So you've heard all about the cool open-source robot simulator MuJoCo, you've tried out the sample robot models (e.g. in MuJoCo Menagerie), and now you want to simulate **your own** robot in MuJoCo- but how to do it? MuJoCo uses the MJCF XML format for its models, and also supports the URDF format. Although some converters from CAD models directly to URDF models exist, but so far I've found it much easier to just write the

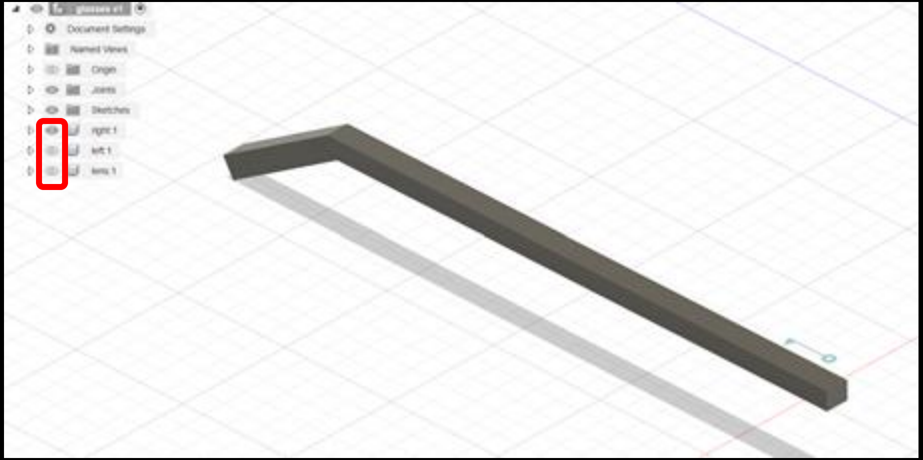
Make the model in your favorite CAD software



Select joint to modify

30.0 deg

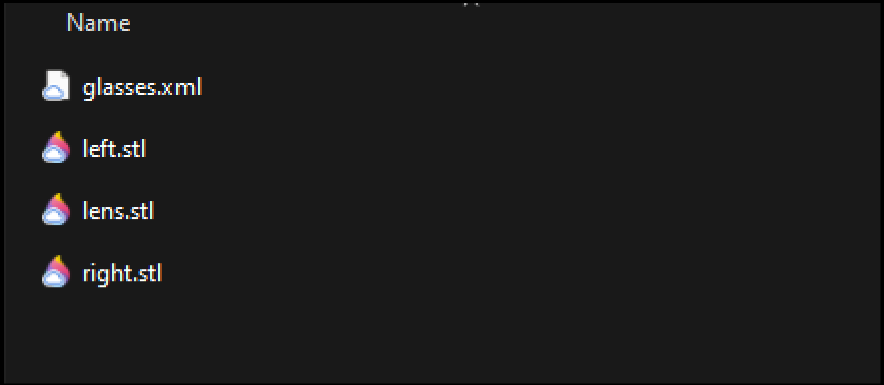
Export STL



I’ve found it much easier if all the STLs are in the same global frame- to do that, (at least for Fusion 360)

1. *Hide* all the components except for the one you want to export
2. Export as STL
3. Repeat for each body

Start writing the XML!



Name

glasses.xml

left.stl

lens.stl

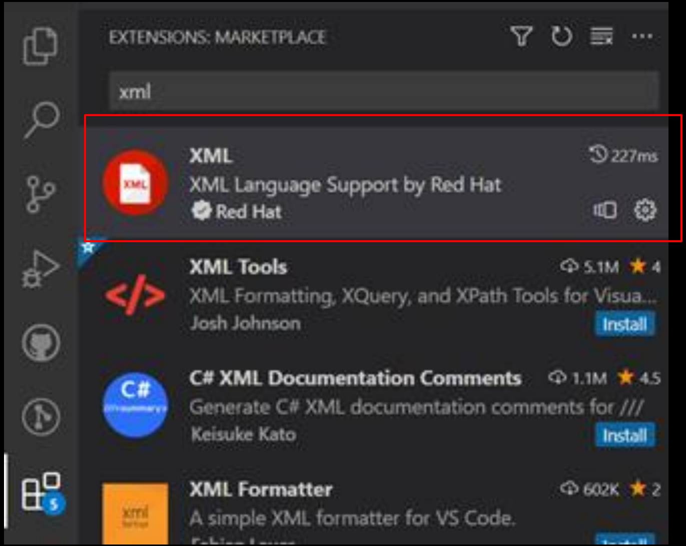
right.stl

- The official documentation is your friend
  - https://mujoco.readthedocs.io/en/latest/XMLreference.html

- Check that your model works in the ./simulate program time to time

- Version control (git) is definitely recommended so you can revert back to the last working state

Some tips for writing XML



xml

XML 227ms
XML Language Support by Red Hat
Red Hat

XML Tools 5.1M ★ 4
XML Formatting, XQuery, and XPath Tools for Visua...
Josh Johnson
Install

C# XML Documentation Comments 1.1M ★ 4.5
Generate C# XML documentation comments for ///
Keisuke Kato
Install

XML Formatter 602K ★ 2
A simple XML formatter for VS Code.

All tags must be properly closed

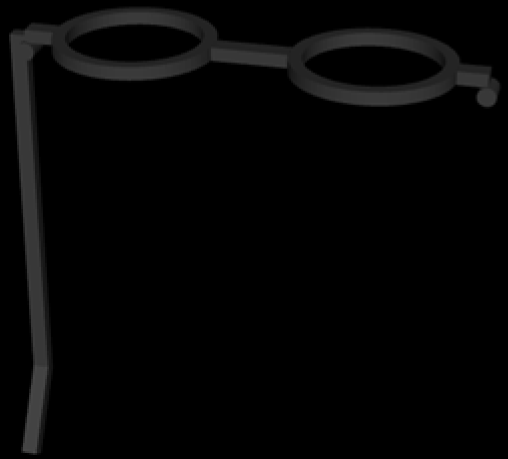
Use indentation to visualize the tag structure

Editor tools can help- for VSCode,

- Install XML extension
- In editor, **ctrl-shift-p** to open commands, and “format with” -> “XML” to apply automatic indentation

Add mesh geometries

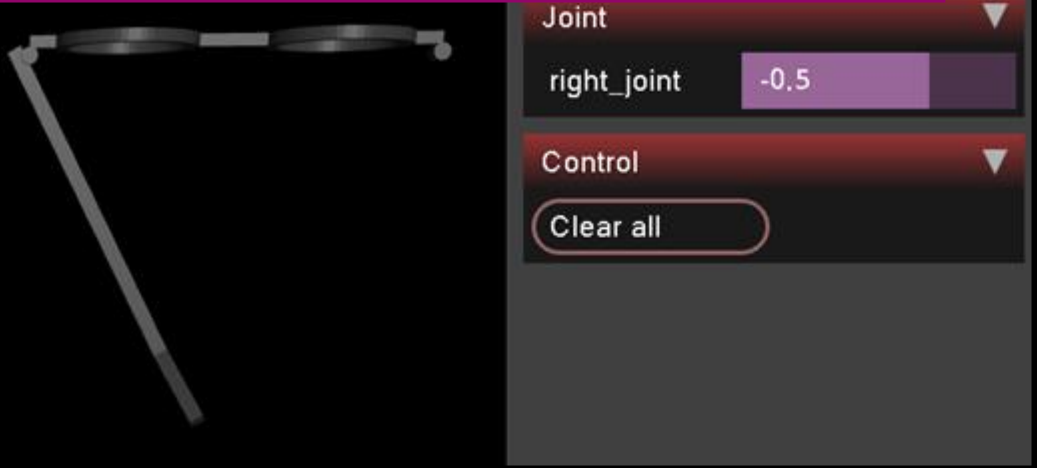
```
<mujoco model="glasses">
    <asset>
        <mesh name="right_mesh" file="right.stl" scale="0.001 0.001 0.001"/>
        <mesh name="lens_mesh" file="lens.stl" scale="0.001 0.001 0.001"/>
    </asset>
    <worldbody>
        <body name="glasses_body">
            <geom type="mesh" mesh="lens_mesh"/>
            <geom type="mesh" mesh="right_mesh"/>
        </body>
    </worldbody>
</mujoco>
```



ETH zürich SoftRobotics Laboratory

Add hinge joint

```
<mujoco model="glasses">
    <asset>
        <mesh name="right_mesh" file="right.stl" scale="0.001 0.001 0.001"/>
        <mesh name="lens_mesh" file="lens.stl" scale="0.001 0.001 0.001"/>
    </asset>
    <worldbody>
        <body name="glasses_body">
            <geom type="mesh" mesh="lens_mesh"/>
            <body name="right_body">
                <joint name="right_joint" pos="-0.067 0 -0.0043" axis="0 1 0" limited="true" damping="0.01" range="-90 0"/>
                <geom type="mesh" mesh="right_mesh"/>
            </body>
        </body>
    </worldbody>
</mujoco>
```



right_joint -0.5

Add actuators

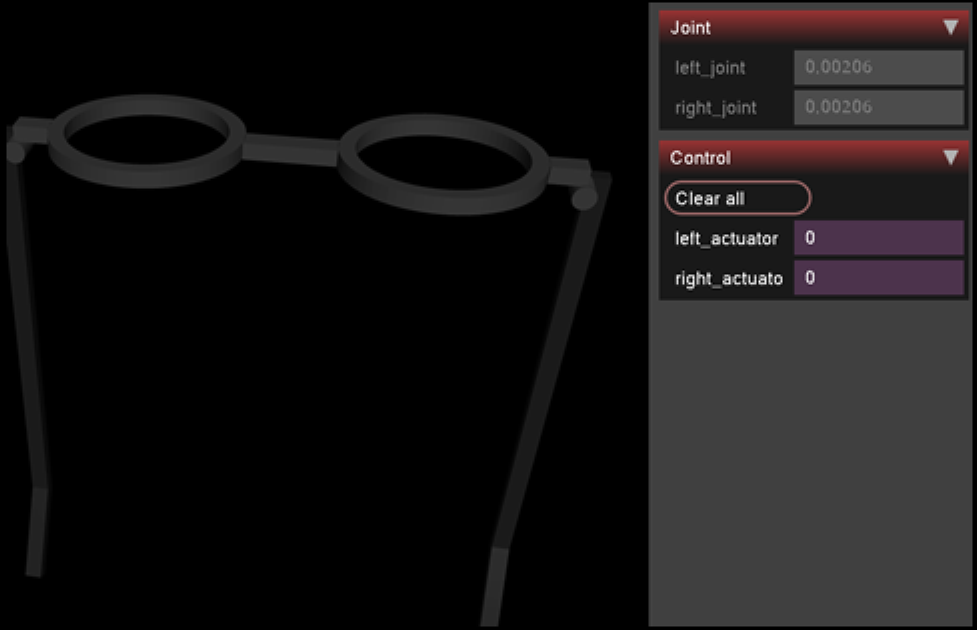
```
<mujoco model="glasses">
    <asset>
        <mesh name="right_mesh" file="right.stl" scale="0.001 0.001 0.001"/>
        <mesh name="lens_mesh" file="lens.stl" scale="0.001 0.001 0.001"/>
    </asset>
    <worldbody>
        <body name="glasses_body">
            <geom type="mesh" mesh="lens_mesh"/>
            <body name="right_body">
                <joint name="right_joint" pos="-0.067 0 -0.0043" axis="0 1 0" limited="true" range="-90 0" damping="0.01"/>
                <geom type="mesh" mesh="right_mesh"/>
            </body>
        </body>
    </worldbody>
    <actuator>
        <position name="right_actuator" joint="right_joint" ctrllimited="true" ctrlrange="-1.57 0" kp="0.1"/>
    </actuator>
</mujoco>
```

Add the left side as well, and use the <default> tag to clean up redundancies

```
<mujoco model="glasses">
    <compiler angle="radian"/>
    <option collision="predefined"/>
    <default>
        <position ctrllimited='true' ctrlrange='0 1.57' kp="0.1"/>
        <joint type="hinge" limited='true' axis='0 1 0' range="0 1.57" damping="0.01"/>
        <geom type="mesh"/>
    </default>

    <asset>
        <mesh name="right_mesh" file="right.stl" scale="0.001 0.001 0.001"/>
        <mesh name="left_mesh" file="left.stl" scale="0.001 0.001 0.001"/>
        <mesh name="lens_mesh" file="lens.stl" scale="0.001 0.001 0.001"/>
    </asset>

    <worldbody>
        <body name="glasses_body">
            <geom mesh="lens_mesh"/>
            <body name="left_body">
                <joint name="left_joint" pos="0.067 0 -0.0043" axis="0 1 0"/>
                <geom mesh="left_mesh"/>
            </body>
            <body name="right_body">
                <joint name="right_joint" pos="-0.067 0 -0.0043" axis="0 -1 0"/>
                <geom mesh="right_mesh"/>
            </body>
        </body>
    </worldbody>
    <actuator>
        <position name="left_actuator" joint="left_joint"/>
        <position name="right_actuator" joint="right_joint"/>
    </actuator>
</mujoco>
```



Joint

left_joint 0.00206

right_joint 0.00206

Control

left_actuator 0

right_actuato 0

ETH zürich

SoftRobotics Laboratory

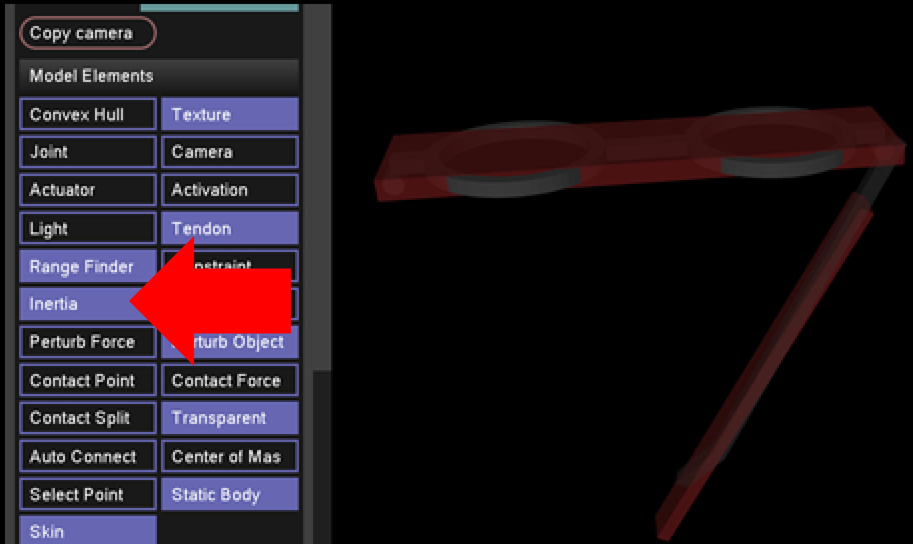
Check the mass

https://mujoco.readthedocs.io/en/latest/XMLreference.html#body-geom

By default, the mass will be calculated from the geom assuming density of water

I recommend setting the mass in the <geom> to the actual measured mass- then the rotational moment of inertia will automatically be calculated based on the geom’s shape.

Visualizing the inertia will show the inertia box, helping you verify the mass settings.

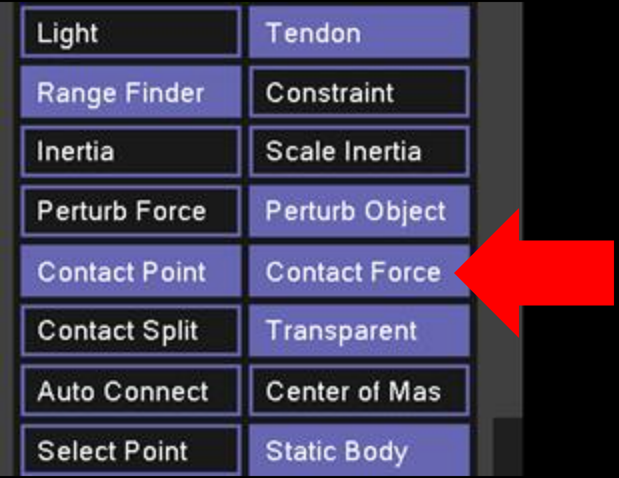


Check for contacts

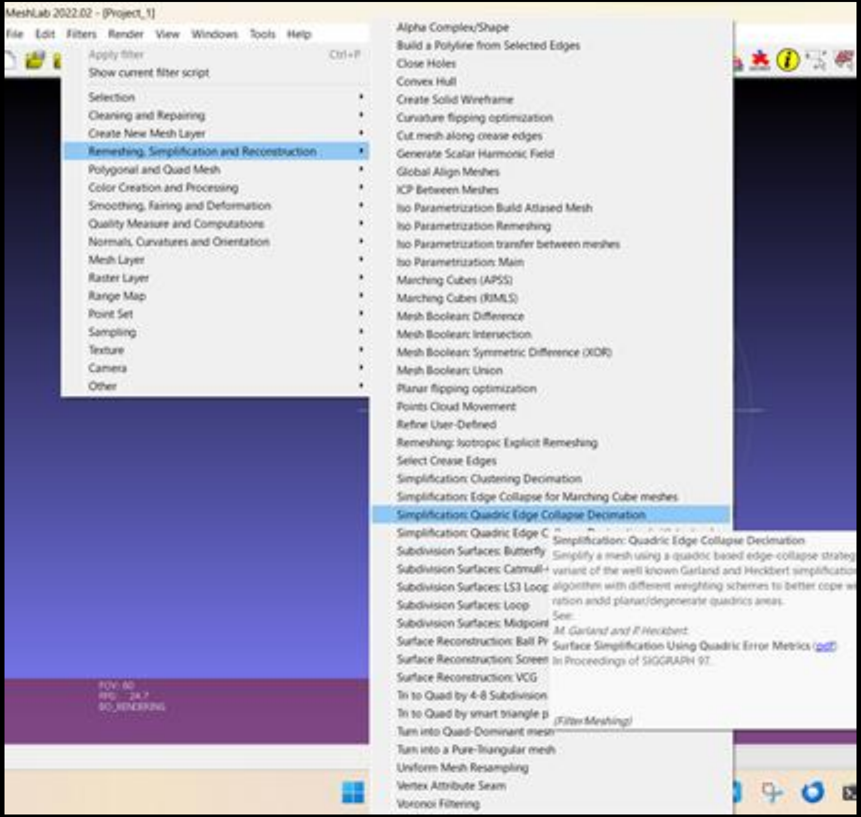
Check if there are no interferences between neighboring bodies

Or, you can just choose to ignore all contact with `<option collision="predefined"/>`

(this setting was deprecated in MuJoCo 3; use https://mujoco.readthedocs.io/en/latest/XMLreference.html#option-flag-contact instead)



If the mesh is too large…


MeshLab 2022.02 - [Project_1]
File
Edit
Filters
Render
View
Windows
Tools
Help
Apply filter
Ctrl+P
Show current filter script
Selection
Cleaning and Repairing
Create New Mesh Layer
Remeshing, Simplification and Reconstruction
Polygonal and Quad Mesh
Color Creation and Processing
Smoothing, Fairing and Deformation
Quality Measure and Computations
Normals, Curvatures and Orientation
Mesh Layer
Raster Layer
Range Map
Point Set
Sampling
Texture
Camera
Other
Alpha Complex/Shape
Build a Polyline from Selected Edges
Close Holes
Convex Hull
Create Solid Wireframe
Curvature flipping optimization
Cut mesh along crease edges
Generate Scalar Harmonic Field
Global Align Meshes
ICP Between Meshes
Iso Parametrization Build Atlased Mesh
Iso Parametrization Remeshing
Iso Parametrization transfer between meshes
Iso Parametrization Main
Marching Cubes (APSS)
Marching Cubes (RIMLS)
Mesh Boolean: Difference
Mesh Boolean: Intersection
Mesh Boolean: Symmetric Difference (XOR)
Mesh Boolean: Union
Planar flipping optimization
Points Cloud Movement
Refine User-Defined
Remeshing: Isotropic Explicit Remeshing
Select Crease Edges
Simplification: Clustering Decimation
Simplification: Edge Collapse for Marching Cube meshes
Simplification: Quadric Edge Collapse Decimation
Simplification: Quadric Edge C
Subdivision Surfaces: Butterfly
Subdivision Surfaces: Catmull-
Subdivision Surfaces: LS3 Loop
Subdivision Surfaces: Loop
Subdivision Surfaces: Midpoint
Surface Reconstruction: Ball Pi
Surface Reconstruction: Screen
Surface Reconstruction: VCG
Tri to Quad by 4-8 Subdivision
Tri to Quad by smart triangle p
Turn into Quad-Dominant mesh
Turn into a Pure-Triangular mesh
Uniform Mesh Resampling
Vertex Attribute Seam
Voronoi Filtering
FOV: 60
FPS: 24.7
BO_RENDERING

Simplification: Quadric Edge Collapse Decimation

Simplify a mesh using a quadric based edge-collapse strateg variant of the well known Garland and Heckbert simplificatio algorithm with different weighting schemes to better cope w ration andd planar/degenerate quadrics areas.

See:

M. Garland and P. Heckbert.

Surface Simplification Using Quadric Error Metrics (pdf)

In Proceedings of SIGGRAPH 97.

(FilterMeshing)

Reduce the number of faces in meshlab

https://www.meshlab.net/

ETH zürich

SoftRobotics Laboratory

Some considerations to make it compatible with IsaacGym

IsaacGym supports MJCF, but their implementation is half-baked and **not all models that work in MuJoCo will work in IsaacGym.**

→ for a sneak peek of user reported issues: https://forums.developer.nvidia.com/search?q=MJCF

Here are some things that we found won't work in IsaacGym (there may be more)

- Spatial tendons (like the ones used in tendon_arm/arm26.xml)
- Collisions can only be calculated for STL meshes (and primitive geoms) and not other mesh types?
- When multiple joints are in the same body, the order in the MJCF file may not be respected (or it may even be completely ignored)

Your task: simulate your robot in MuJoCo

Must definitely be done before Unit 8, where you will load the model into IsaacGym

If in doubt about what parameters to set, refer to the Shadow Hand model and Faive Hand model

Sample model files

https://github.com/srl-ethz/faive_gym_oss/tree/main/assets/faive_hand_p0 : Faive Hand model

https://github.com/NVIDIA-Omniverse/IsaacGymEnvs/tree/main/assets/mjcf : sample MJCF models used in IsaacGymEnvs

The above two models will definitely work in IsaacGym

https://github.com/google-deepmind/mujoco_menagerie : various high-quality MJCF files