Articulated bodies

The SOFA team

2007

Abstract

This document explains the principles of the different types of articulated bodies implemented in SOFA. For each kind of articulated body, it first explains quickly the concepts of the method, then shows how to use it in a scene, and finally deals with the implementation.

Contents

1 Soft Articulations

1.1 Concepts

The objective of this method is to use stiff forces to simulate joint articulations, instead of classical constraints.

To do this, a joint is modeled by a 6 degrees of freedom spring. By the way, the user specify a stiffness on each translation and rotation.

- A null stiffness defines a free movement.

- A huge stiffness defines a forbidden movement.

- All nuances are possible to define semi constrained movements.

2 main advantages can be extracted from this method :

- A better stability. As we don't try to statisfy constraints but only apply forces, there is always a solution to resolve the system.

- more possibilities to model articulations are allowed. As the stiffnesses define the degrees of freedom of the articulations, a better accuracy is posssible to simulate free movements as forbidden movements, i.e. an articulation axis is not inevitably totally free or totally fixed.

1.2 Realization

To define physically an articulated body, we first have a set of rigids (the bones). *cf fig. 1*
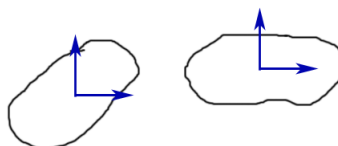

Figure 1: two bones

Each of these bones contains several articulations points, also defined by rigids to have orientation information. *cf fig. 2*
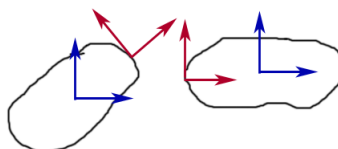

Figure 2: two bones (blue) with their articulation frames (red)

As seen previously, a joint between 2 bones is modeled by a 6-DOF spring. These springs are attached on the articulations points. *cf fig. 3*
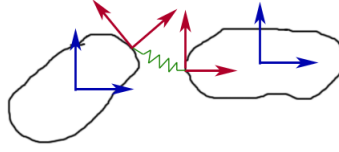
Figure 3: two bones linked by a joint-spring

1.3 Sofa implementation

To simulate these components in Sofa, we first need 2 mechanical objects : one for the bones (independent DOFs), and an other for the articulation points (mapped DOFs). Each of them contains a list of rigid DOFs (respectively all the bones and all the articulations of the articulated body). A mapping performs the link between the two lists, to know which articulations belong to which bones.

1.3.1 Corresponding scene graph

Figure 4: a simple articulated body scene

1.3.2 Example

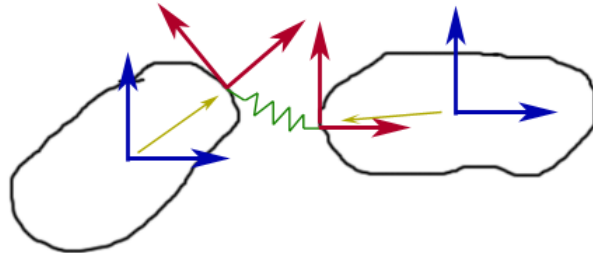The example softArticulations.scn shows a basic pendulum :

```
<Node>
  <Object type="BruteForceDetection"/>
  <Object type="DefaultContactManager"/>
  <Object type="DefaultPipeline"/>
```

```
<Object type="ProximityIntersection"/>

<Node>
  <Object type="CGImplicitSolver" />
  <Object type="MechanicalObject" template="Rigid" name="bones DOFs"
          position="0 0 0   0 0 0 1
                    1 0 0   0 0 0 1
                    3 0 0   0 0 0 1
                    5 0 0   0 0 0 1
                    7 0 0   0 0 0 1" />
  <Object type="UniformMass" template="Rigid" name="bones mass"
          mass="1 1 [1 0 0,0 1 0,0 0 1]" />
  <Object type="FixedConstraint" template="Rigid" name="fixOrigin"
          indices="0" />

  <Node>
    <Object type="MechanicalObject" template="Rigid" name="articulation points"
            position="0 0 0   0.707914 0 0 0.707914
                     -1 0 0   0.707914 0 0 0.707914
                      1 0 0   0.707914 0 0 0.707914
                     -1 0 0   0.707914 0 0 0.707914
                      1 0 0   0.707914 0 0 0.707914
                     -1 0 0   0.707914 0 0 0.707914
                      1 0 0   0.707914 0 0 0.707914
                     -1 0 0   0.707914 0 0 0.707914
                      1 0 0   0.707914 0 0 0.707914" />
    <Object type="RigidRigidMapping"
            repartition="1 2 2 2 2" />
    <Object type="JointSpringForceField" template="Rigid" name="joint springs"
            spring="0 1   0 0 0 0 1 0   0 30000 0 200000   0  0 0 0  0 0 0 1
                    2 3   0 0 0 0 1 0   0 30000 0 200000   0  0 0 0  0 0 0 1
                    4 5   0 0 0 0 1 0   0 30000 0 200000   0  0 0 0  0 0 0 1
                    6 7   0 0 0 0 1 0   0 30000 0 200000   0  0 0 0  0 0 0 1" />
  </Node>
  <Node>
    <Object type="MechanicalObject" template="Vec3d"
            position="-1 -0.5 -0.5   -1 0.5 -0.5 ..." />
    <Object type="MeshTopology"
            lines="0 1   1 2   ..."
            triangles="3 1 0   3 2 1   ..." />
    <Object type="TriangleModel"/>
    <Object type="LineModel"/>
    <Object type="RigidMapping"
            repartition="0 8 8 8 8" />
  </Node>
</Node>
</Node>
```

Rigid fixed by ⟶
a constraint

Rigids modeling the bones
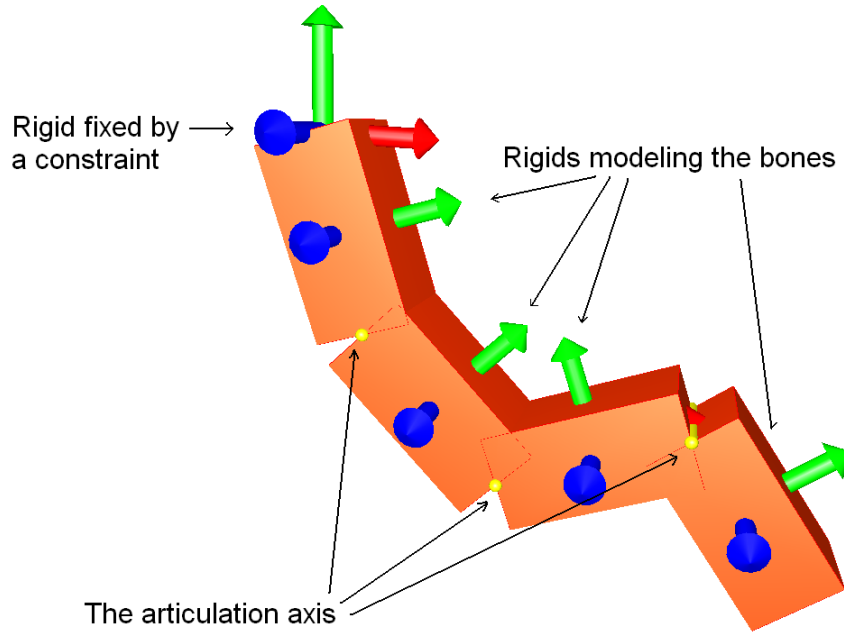
The articulation axis

Figure 5: The pendulum is composed by 4 rigids linked one by one by articulations

In this example, we have under the first node the components to manage collisions, as usual. Under the second node, we have :

- the solver,

- the mechanical object modeling the independent rigid DOFs (5 rigids here),

- the rigid mass,

- a constraint, to fix the first rigid.

The third node (a child of the previous one) contains the components relative to the articulations :

- the mechanical object modeling articulation points. Positions and orientations are relative to their parents.

- the mapping to link the two mechanical objects, as explained before. To know which articulations belong to which bones, a repartition vector is used. Several cases for this vector are possible :

  - no value specified : every articulations belong to the first bone (classic rigid mapping).
  - one value specified (ex: repartition="2") : each bone has the same number of articulations.
  - number of bones values (like here, repartition="1 2 2 2") : the number of articulations is specified for each bone. For instance, here the first bone has 1 articulation, the next has 2 articulations, the next 2, Etc.

- the JointSpringForceField containing the springs (4 springs here). Each spring is defined by a list of parameters. For instance for the first spring we have "0 1 0 0 0 0 1 0 0 30000 0 200000 0 0 0 0 0 0 0 1".

- "0 1" are the indices of the two articulations the spring is attached to
- "0 0 0 0 1 0" design the free axis for the movements. "0 0 0" mean that the 3 translation axis are constrained, and "0 1 0" mean that only the Y rotation axis is free.
- "0 30000 0 2000000" are the stiffnesses for each kind of movement: "0 30000" are respectively for free translation and for constrained translation", and "0 2000000" are respectively for free rotation and for constrained rotation.
- "0" is the damping factor
- "0 0 0" is to specify the initial translation
- "0 0 0 1" is to specify the initial rotation (quaternion)

The last node contains the collision model. Nothing special here.

1.4 Skinning

The articulated body described previously models the skeleton of an object. To have the external model (for the visual model or the collision model), which follows correctly the skeleton movements, it has to be mapped with the skeleton. A skinning mapping allows us to do this link. The external model is from this moment able to deform itself smoothly, i.e. without breaking points around the articulations.

The influence of the bones on each point of the external model is given by skinning weights. 2 ways are possible to set the skinning weights to the mapping :

- Either the user gives directly the weights list to the mapping. It is useful if good weights have been pre computed previsouly, like in Maya for instance.

- Else, the user defines a number of references n that will be used for mapped points. Then, each external model point will search its n nearest bones (mechanical DOFs), and then compute the skinning weights from the relation :

$$W = \frac{1}{d^2}$$

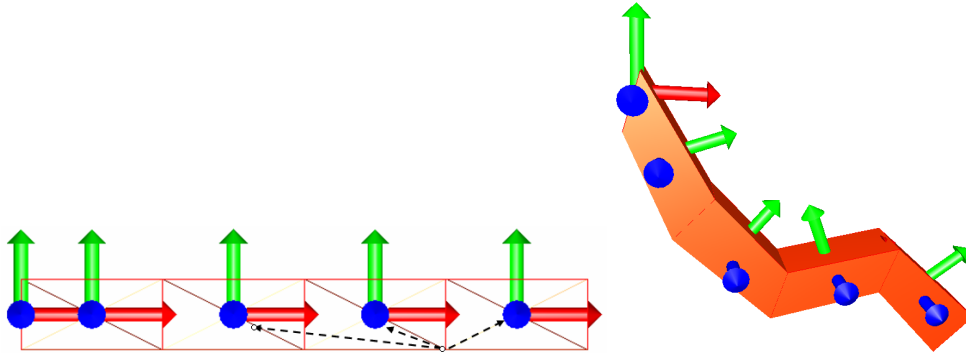with d : the distance between the external point and the rigid DOF.

Figure 6: In the example "softArticulationsSkinned.scn" the external points compute their skinning weights from the 3 nearest DOFs

2 Articulated System Description

2.1 Concepts

The main idea is to place the description of an articulated system of rigid bodies in the scene graph, so that it can be easily visualized, created and modified. For now, only serial articulated system are considered : it is considered as a hierarchy of 6 dofs solids, starting from a root solid that is fixed by default. (However, it can be animated using *multimapping* approach see**??**).

The main hypothesis of the Articulated System Description is that the position, velocity and forces (i.e. the mechanical state) of the rigid bodies that are animated with the articulated system are described in the same state with rigid types.

2.2 Realization

The description of the articulated system is composed of two parts:

- The **articulation centers** provide the point of articulation between two 6D bodies. It is described by its local position on the two bodies: the parent and the child. Its position on the global frame can be deduced from the position in the global frame of the parent.
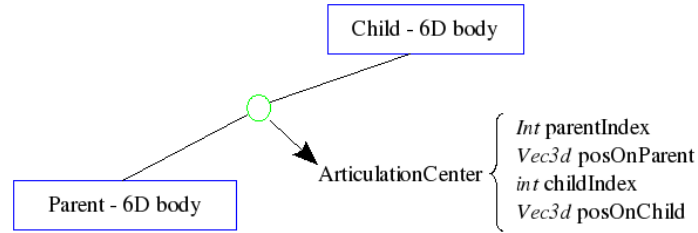
Figure 7: articulation center

- Each articulation center has a set of **articulations** that defines the degrees of freedom of the child on its parent. By default, if no articulation are defined, the position of the child is directly deduced from the position of the parent. Articulation can be translation or rotation and are defined by an axis. Each articulation represent one degree of freedom, and the axis of each articulation are updated sequentially.
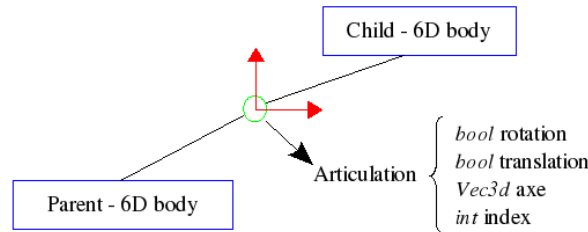
Figure 8: articulation

This representation is close to the one use in BVH format. BVH files can be imported in sofa in order to create an Articulated System. An example of an import can be found in the scene

/examples/Components/mapping/ArticulatedSystemMapping_ bvh.scn.

Note that the BVH format also include an animation sequence for each articulation. The values at each frame are re-interpolated in a ArticulatedHierarchyBVHController in order to follow the time step defined in the simulation. This controller provides the position of each articulation in a Vec1 type vector. The motion of the hole articulated system is computed thanks to a mapping described in the following.

2.2.1 Articulated System Mapping

The degrees of freedom of the articulated system are given in a Vec1 type vectors containing the position, velocity and force at each articulation. As the positions of the rigid bodies can be deduced from the position of the articulations, the articulation system can be described in a mapping. In the scene graph, we obtain:
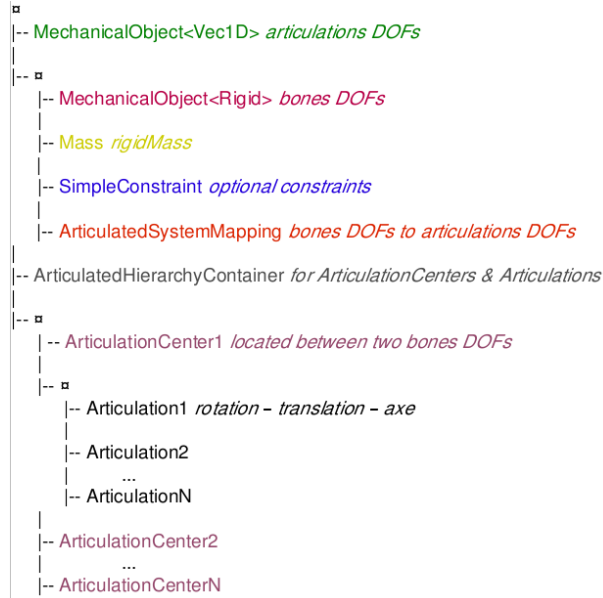

Figure 9: articulated system graph

2.2.2 Example

The example */examples/Components/mapping/ArticulatedSystemMapping.scn* shows a basic pendulum :

```
1  <Object type="BruteForceDetection"/>
2  <Object type="DefaultContactManager"/>
3  <Object type="DefaultPipeline"/>
4  <Object type="ProximityIntersection"/>
5
6  <Node>
7    <Object type="EulerImplicit" name="cg odesolver" printLog="false"/>
8    <Object type="CGLinearSolver" iterations="100" name="linear solver" threshold="1e-20"
          tolerance="1e-20"/>
9    <Object type="MechanicalObject" template="Vec1d" name="Articulations"
          position="0 0 0 0"/>
11   <Node>
12     <Object type="MechanicalObject" template="Rigid" name="DOFs"
```

```xml
13              position="0 0 0 0 0 0 1
14                       1 0 0  0 0 0 1
15                       3 0 0  0 0 0 1
16                       5 0 0  0 0 0 1
17                       7 0 0  0 0 0 1"/>
18      <Object type="UniformMass" template="Rigid" name="mass"
19              mass="1 1 [1 0 0,0 1 0,0 0 1]"/>
20      <Object type="FixedConstraint" template="Rigid" name="fixOrigin"
21              indices="0"/>
22      <Object type="ArticulatedSystemMapping"/>
23    </Node>
24    <Object type="ArticulatedHierarchyContainer"/>
25    <Node name="articulationCenters">
26      <Node name="articulationCenter1">
27        <Object type="ArticulationCenter"
28                parentIndex="0"
29                childIndex="1"
30                posOnParent="0 0 0"
31                posOnChild="-1 0 0"/>
32        <Node name="articulations">
33          <Object type="Articulation"
34                  translation="0"
35                  rotation="1"
36                  rotationAxis="0 0 1"
37                  articulationIndex="0"/>
38        </Node>
39      </Node>
40      <Node name="articulationCenter2">
41        <Object type="ArticulationCenter"
42                parentIndex="1"
43                childIndex="2"
44                posOnParent="1 0 0"
45                posOnChild="-1 0 0"/>
46        <Node name="articulations">
47          <Object type="Articulation"
48                  translation="0"
49                  rotation="1"
50                  rotationAxis="0 0 1"
51                  articulationIndex="1"/>
52        </Node>
53      </Node>
54      <Node name="articulationCenter3">
55        <Object type="ArticulationCenter"
56                parentIndex="2"
57                childIndex="3"
58                posOnParent="1 0 0"
59                posOnChild="-1 0 0"/>
60        <Node name="articulations">
61          <Object type="Articulation"
62                  translation="0"
63                  rotation="1"
64                  rotationAxis="0 0 1"
65                  articulationIndex="2"/>
66        </Node>
67      </Node>
68      <Node name="articulationCenter4">
69        <Object type="ArticulationCenter"
70                parentIndex="3"
71                childIndex="4"
72                posOnParent="1 0 0"
73                posOnChild="-1 0 0"/>
74        <Node name="articulations">
75          <Object type="Articulation"
76                  translation="0"
77                  rotation="1"
78                  rotationAxis="0 0 1"
79                  articulationIndex="3"/>
80        </Node>
```

```
81      </Node>
82    </Node>
83  </Node>
```

In this example, we have under the first node the components to manage collisions, as usual. Under the second node, we have :

- the solver,

- the mechanical object modeling the position of each articulation (4 articulations here),

- the container to store all the articulation data.

The third node (a child of the previous one) contains the components relative to the articulations :

- the mechanical object modeling the independent rigid DOFs (5 rigids here),

- the rigid mass,

- a constraint, to fix the first rigid.

- the mapping to link the two mechanical objects, as explained before.