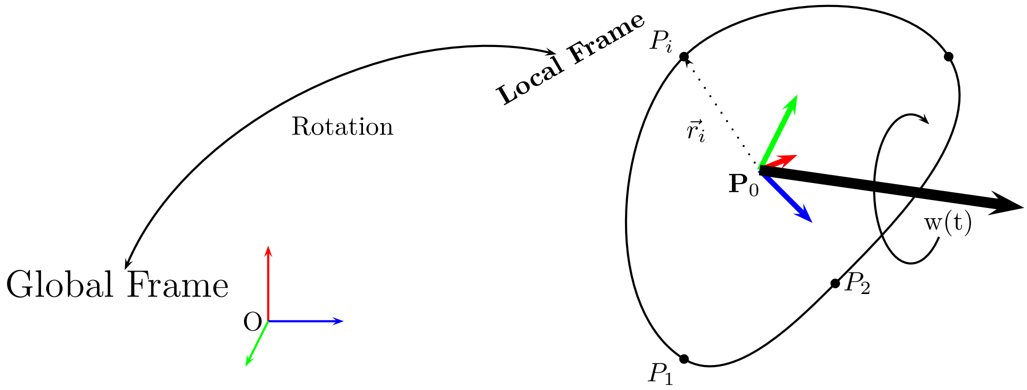
Rigid Mapping

Motivation: A rigid body (perfectly rigid) is an undeformable body. Computation for simulating its motion and colision is simpler than a deformable one. To optimize the computation, Rigid bodies in Sofa are presented by different models (Visual Model, Colision Model, Behaviour Model) ensuring their appropriate task. These model are related by the mapping (rigid mapping in this case).



Definitions:

- Model Behaviour Rigid : Position and orientation (quaternion) of the center(s) of mass.
- Model Colision Rigid : A set of particles used for computing the colision with other objects.
- Model Visual Rigid : A set of particle used for visualizing.

Note that a rigid object can contain several rigid bodies (ex articulation system), consequently several center of mass. In the simple case, there is only one center.

Computation: By its proprety undeformable, rigid body composed a set of particles P_i ensuring the distance of any pair of point and the angle of any pair of vectors remains constant in relation to time. Two frames of reference are needed to descript a rigid body : global and local. To find the position of any particles, only one particle is needed to be found in the global frames, then the other ones are related on the local frame. The center of mass is chosen naturally for the local frame origin P_o. After one step of time, the rigid position (linear and angular) can be changed. The mapping update all other particles potition and velocity knowing only these of the updated center. If the rigid is in colision, several particles are in solicitation. The mapping compute how are solicited the center of rigid. Assumed a rigid body has N particles : $P_{i\{i=0...N\}}$, their coordinates in the local frame are r_i. These works of mapping are descripted by those following algorithms :

To update the particles position :

Finding $OP_o = LinearPosition;$
Finding $R_{loc}^{glob} = RotationMatrix;$
For$(i = 1; i < N; i++)$
{
$P_oP_i = R_{loc}^{glob} * r_i;$
$OP_i = OP_o + P_oP_i;$
}

To update the particles velocity :

Finding $V_{P_O} = LinearVelocity;$
Finding $w_{Po} = AngularVelocity;$
For$(i = 1; i < N; i++)$
{
$V_i = V_{P_O} - P_oP_i \wedge w_{Po};$
}

To compute the external forces soliciting to center :

For$(i = 1; i < N; i++)$
{
$F_{exter} += f_i;$
$Torque += P_oP_i \wedge f_i$
}
Update $LinearForce = F_{exter};$
Update $AngularForce = Torque;$

Implementated code in Sofa: These algorithms are implemented in Sofa by the following methods :
To update the particles position :

```
template <class BasicMapping>
void RigidMapping<BasicMapping>::apply(typename Out::VecCoord& out, const typename In::VecCoord& in)
```

To update the particles velocity :

```
template <class BasicMapping>
void RigidMapping<BasicMapping>::applyJ(typename Out::VecDeriv& out, const typename In::VecDeriv& in)
```

To compute the external forces soliciting to center :

```
template <class BasicMapping>
void RigidMapping<BasicMapping>::applyJT(typename In::VecDeriv& out, const typename Out::VecDeriv& in)
```

Sofa Keyword: MechanicalState, Bihaviour Model, Colision Model, Visual Model, Mapping, Quaternion.

Document written by
Chi Thanh NGUYEN
INRIA Lille