CS 277 - Experimental Haptics
Lecture 10

Deformable Models (II)

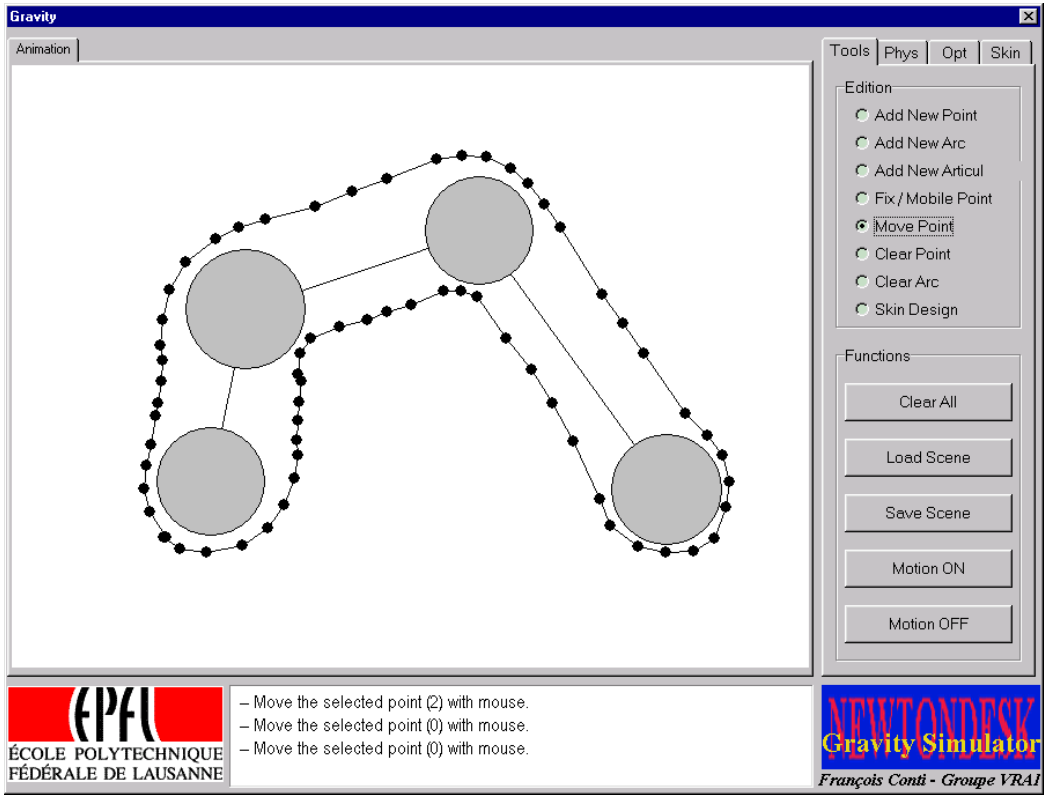
Outline

- Introduction
- Spring Models
- Computing Dynamics in the Haptics Loop
- Filling Sphere Approach for Elastic Models
- Computing Collision Detection in Real Time
- Demonstrations

Outline

- Introduction
- Spring Models
- Computing Dynamics in the Haptics Loop
- Filling Sphere Approach for Elastic Models
- Computing Collision Detection in Real Time
- Demonstrations

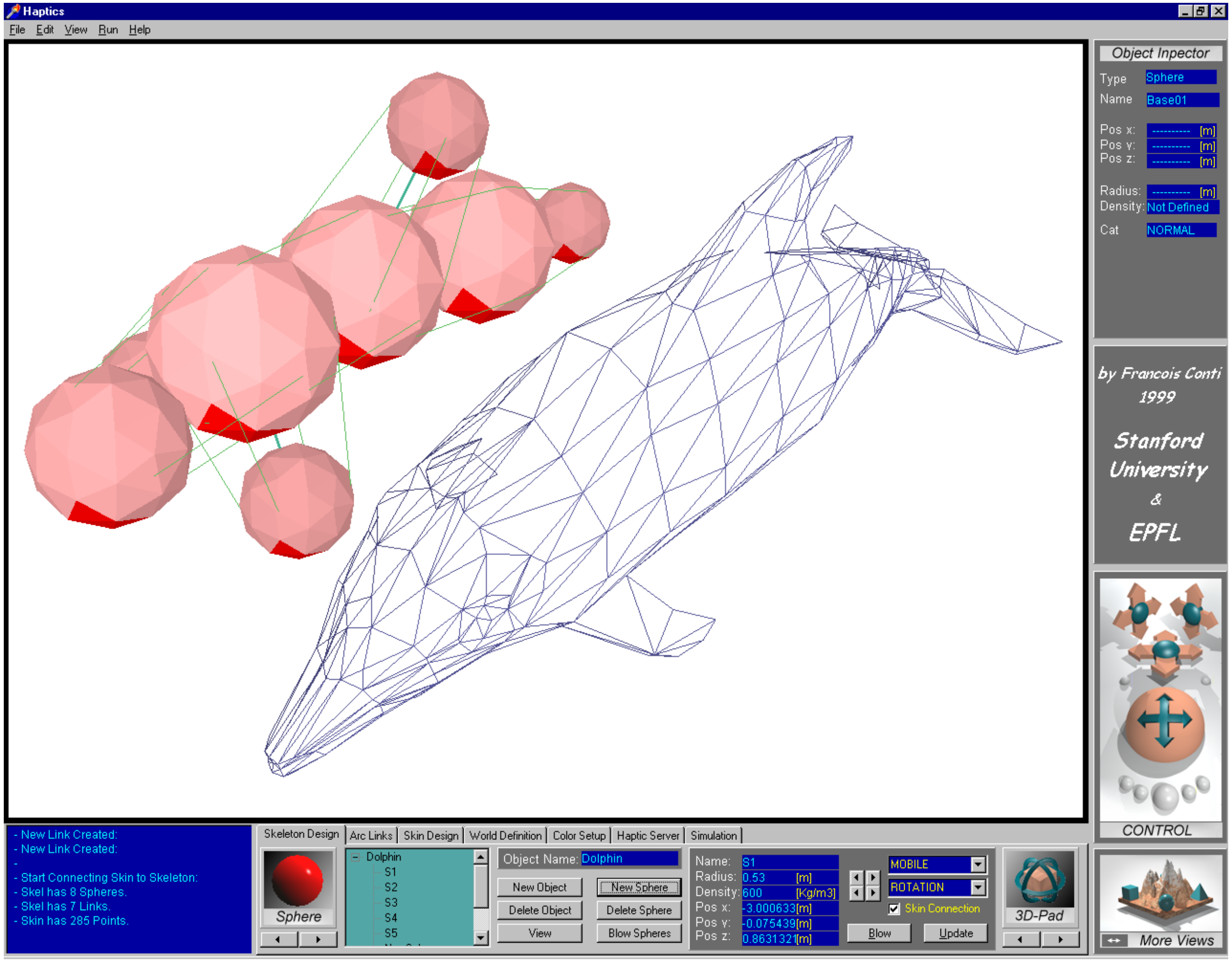
Filling Sphere Approach (2D)



Edition
○ Add New Point
○ Add New Arc
○ Add New Articul
○ Fix / Mobile Point
● Move Point
○ Clear Point
○ Clear Arc
○ Skin Design

– Move the selected point (2) with mouse.
– Move the selected point (0) with mouse.
– Move the selected point (0) with mouse.

Filling Sphere Approach (3D)



Type: Sphere
Name: Base01
Pos x: ---------- [m]
Pos y: ---------- [m]
Pos z: ---------- [m]
Radius: ---------- [m]
Density: Not Defined
Cat: NORMAL

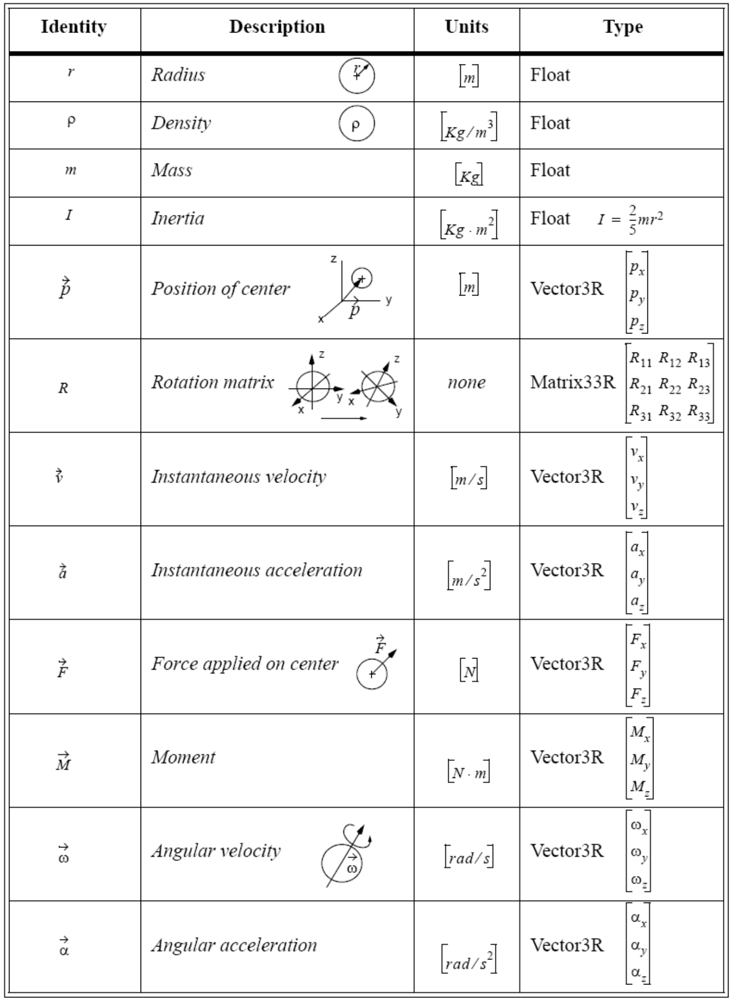
- New Link Created:
- New Link Created:
-
- Start Connecting Skin to Skeleton:
- Skel has 8 Spheres.
- Skel has 7 Links.
- Skin has 285 Points.

Object Name: Dolphin

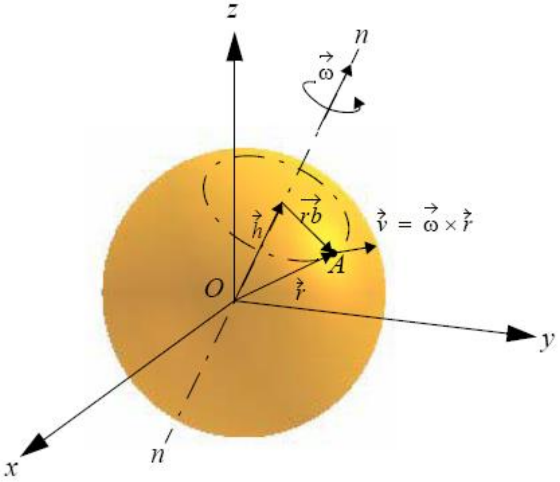
Name: S1
Radius: 0.53 [m]
Density: 600 [Kg/m3]
Pos x: -3.000633[m]
Pos y: -0.075439[m]
Pos z: 0.8631321[m]

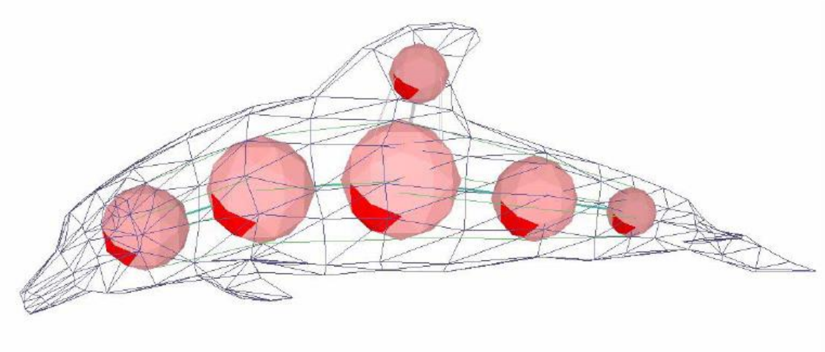
MOBILE
ROTATION
☑ Skin Connection

Mass Nodes

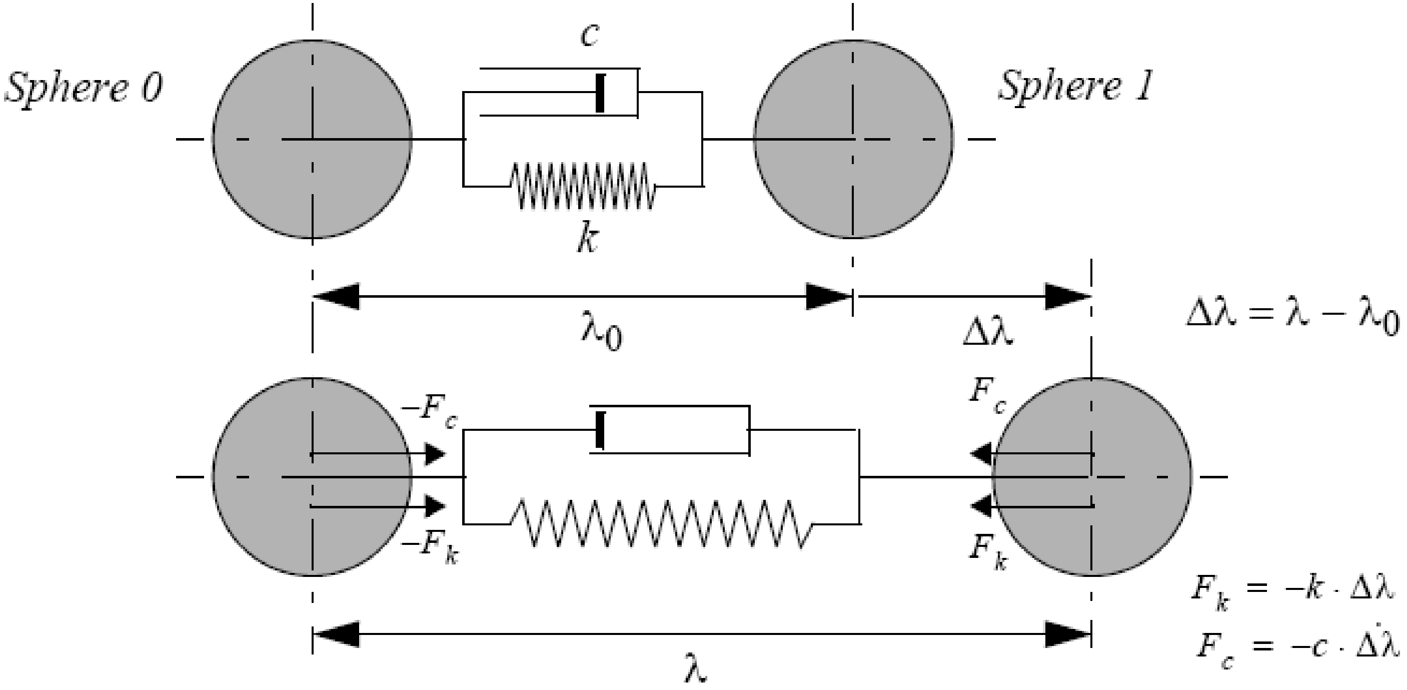


Identity	Description	Units	Type
r	*Radius*	$[m]$	Float
ρ	*Density*	$[Kg/m^3]$	Float
m	*Mass*	$[Kg]$	Float
I	*Inertia*	$[Kg \cdot m^2]$	Float $I = \frac{2}{5}mr^2$
$\vec{p}$	*Position of center*	$[m]$	Vector3R $\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$
R	*Rotation matrix*	*none*	Matrix33R $\begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$
$\vec{v}$	*Instantaneous velocity*	$[m/s]$	Vector3R $\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$
$\vec{a}$	*Instantaneous acceleration*	$[m/s^2]$	Vector3R $\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$
$\vec{F}$	*Force applied on center*	$[N]$	Vector3R $\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix}$
$\vec{M}$	*Moment*	$[N \cdot m]$	Vector3R $\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix}$
$\vec{\omega}$	*Angular velocity*	$[rad/s]$	Vector3R $\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$
$\vec{\alpha}$	*Angular acceleration*	$[rad/s^2]$	Vector3R $\begin{bmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \end{bmatrix}$

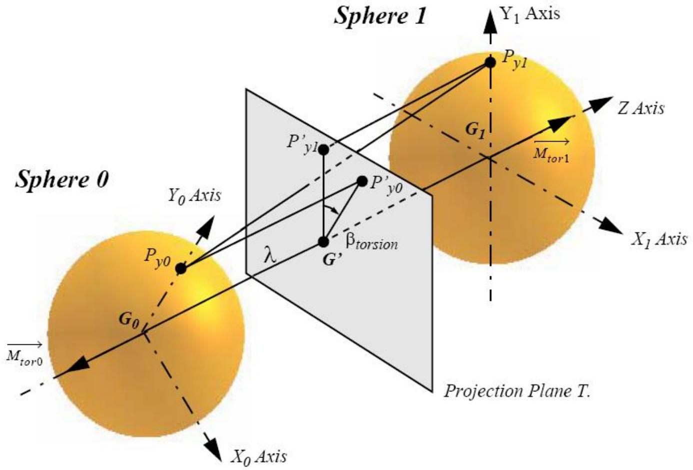


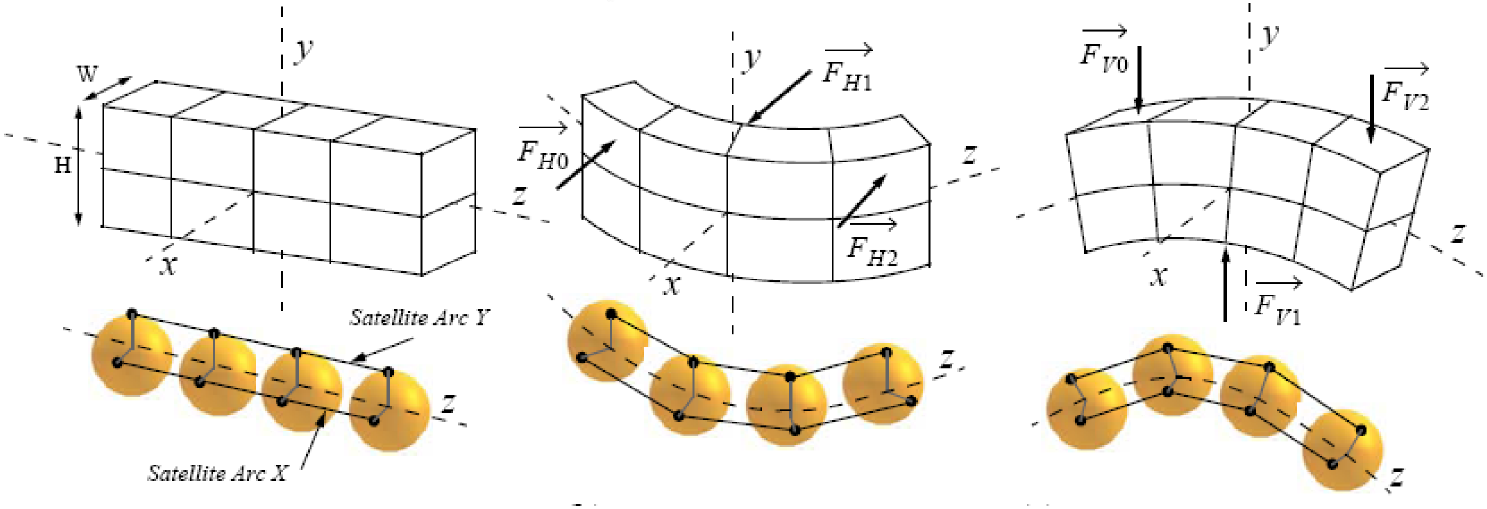


Elongation


Sphere 0
c
Sphere 1
k
λ_0
$\Delta\lambda$
$\Delta\lambda = \lambda - \lambda_0$
$-F_c$
F_c
$-F_k$
F_k
$F_k = -k \cdot \Delta\lambda$
$F_c = -c \cdot \dot{\Delta\lambda}$
λ

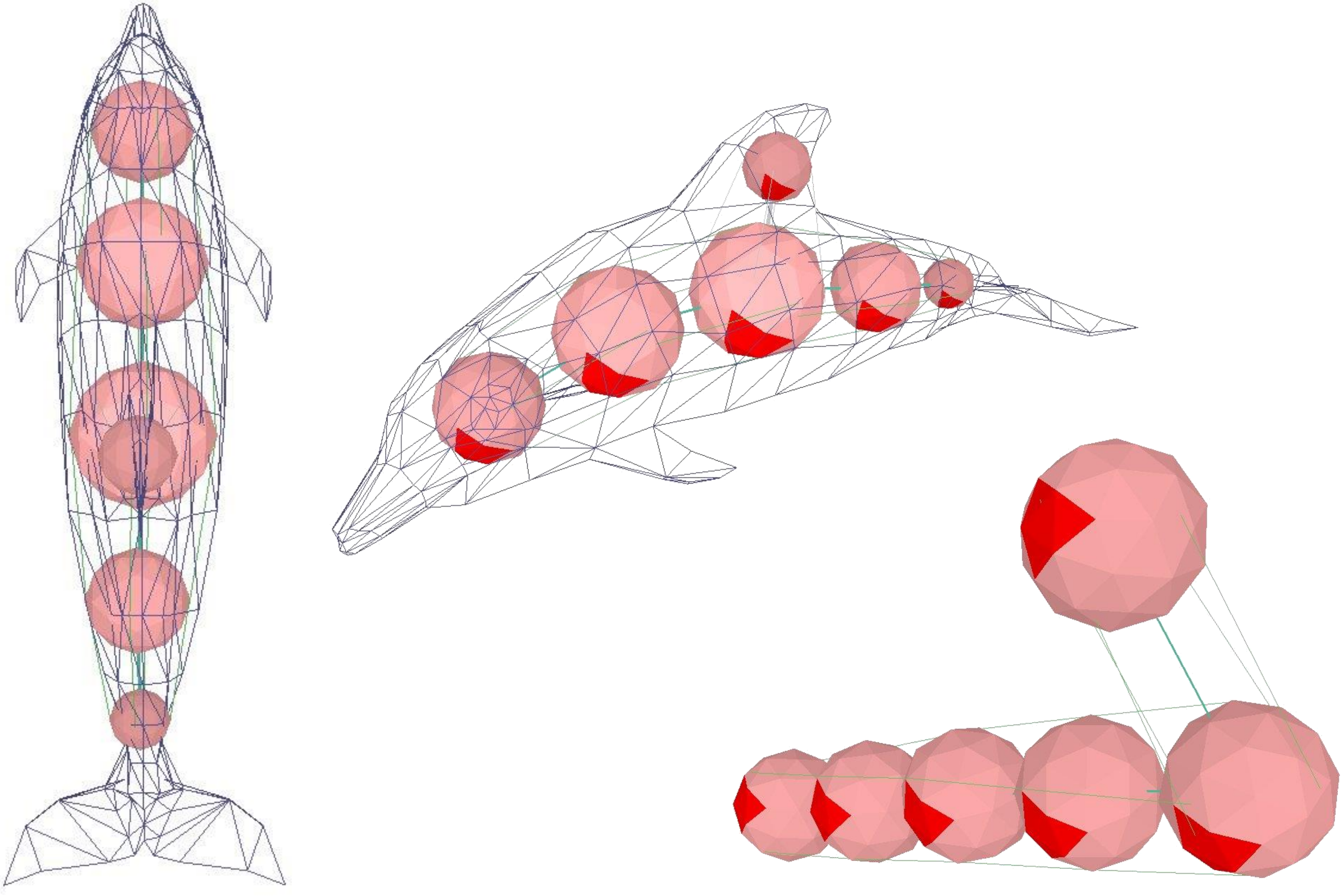
Flexion and Torsion





W
y
H
x
z
Satellite Arc Y
Satellite Arc X
$\overrightarrow{F_{H0}}$
$\overrightarrow{F_{H1}}$
$\overrightarrow{F_{H2}}$
$\overrightarrow{F_{V0}}$
$\overrightarrow{F_{V1}}$
$\overrightarrow{F_{V2}}$

Mesh and Skeleton



Outline

- Introduction
- Spring Models
- Computing Dynamics in the Haptics Loop
- Filling Sphere Approach for Elastic Models
- Computing Collision Detection in Real Time
- Demonstrations

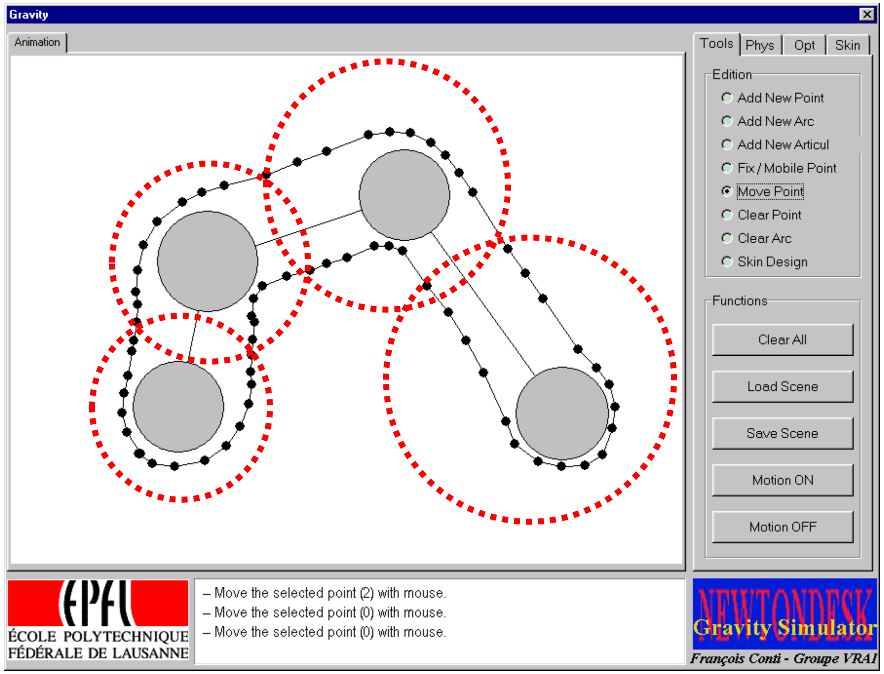
Collision Detection

Collision detection with deformable meshes are difficult to achieve in real time due to the constant change of their geometry (constant update of the collision detection model)

- How can we compute a collision between a segment and deformable mesh?
- How can we compute collisions between deformable meshes?

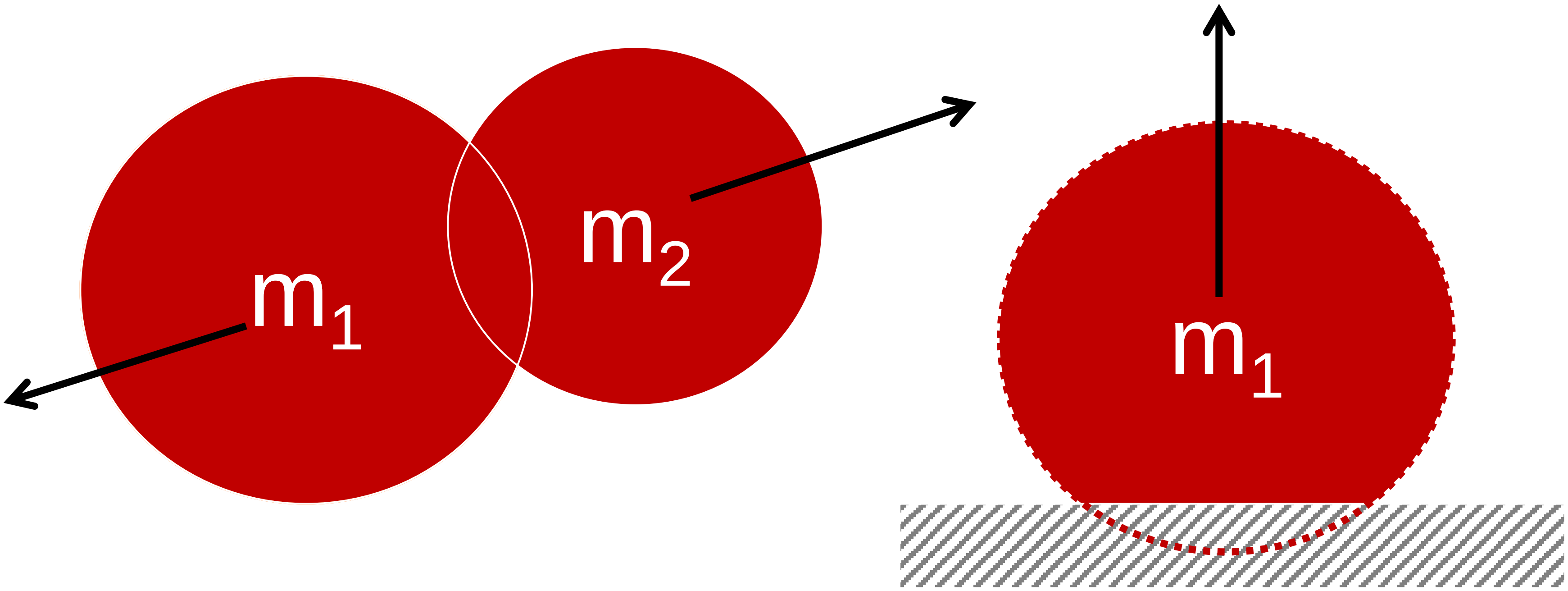
Collision Segment-Mesh

1. Collision detection is first performed between the input segment and the collision spheres composing the skeleton of the model.
2. Collision between the segment and the triangles are then searched locally



– Move the selected point (2) with mouse.
– Move the selected point (0) with mouse.
– Move the selected point (0) with mouse.

Collision Mesh-Mesh



Reaction forces are computed between mass nodes

$F_r = -k\,x$

Outline

- Introduction
- Spring Models
- Computing Dynamics in the Haptics Loop
- Filling Sphere Approach for Elastic Models
- Computing Collision Detection in Real Time
- Demonstrations