



CoppeliaSim — riguardo l'esercizio 1 — prismatic joints

[Lo schema da implementare](#)

[La struttura del prof](#)

[V1 — Alcune osservazioni sulle misure](#)

[Visuale senza il rendering 3d](#)

[Montare correttamente un prismatic joint](#)

[Joints e struttura](#)

[Regolazione del prismatic joint](#)

[Dimensioni](#)

[Posizione e range del pistone](#)

[Proprietà dinamiche](#)

[V2 — prismatic joint bello](#)

[Prima impostazione del joint](#)

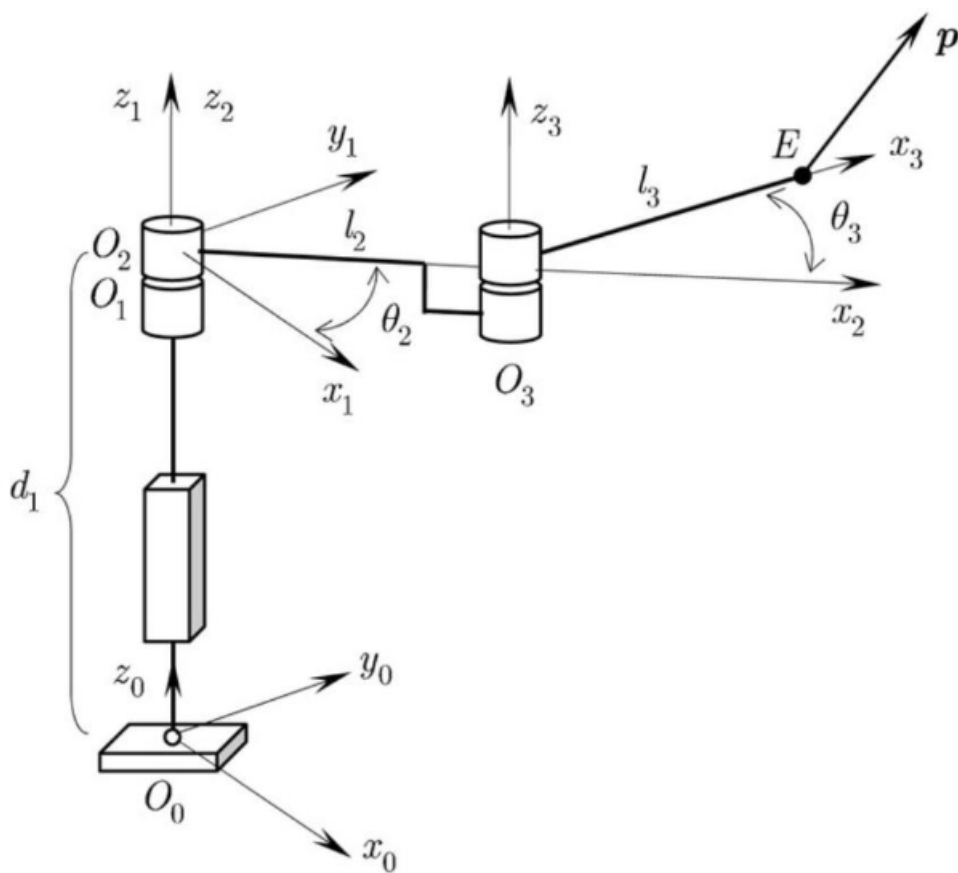
[Lavorare su una mesh](#)

[Problemi di fisica...](#)

[Il risultato](#)

Lo schema da implementare

Realizzazione del seguente schematic:



Con questi constraints:

$d_1 = 0.4 \text{ m}$ (initial length)

$l_2 = 0.25 \text{ m}$

$l_3 = 0.1 \text{ m}$

Joint Limits:

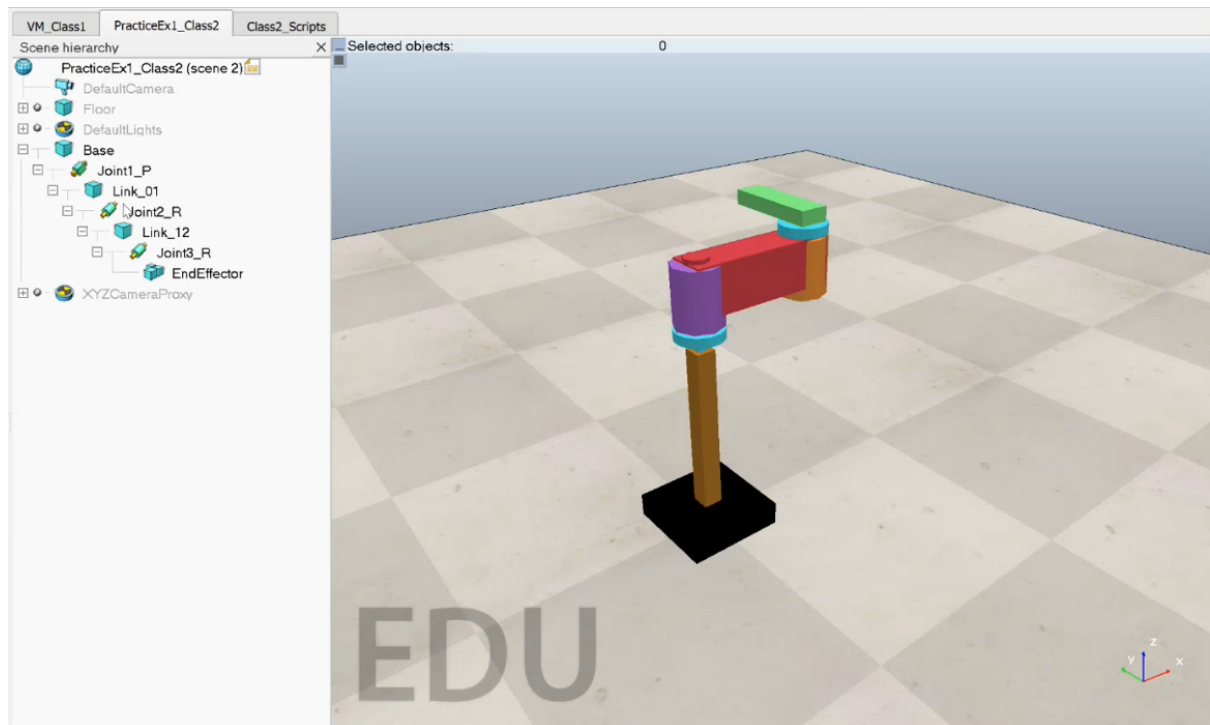
$0.01 \text{ m} < d_1 < 0.5 \text{ m}$

$\theta_1 \rightarrow \text{Cyclic}$

$-120 \text{ deg} < \theta_2 < 120 \text{ deg}$

La struttura del prof

...bruttina... ma funziona. E' possibile fare di meglio?



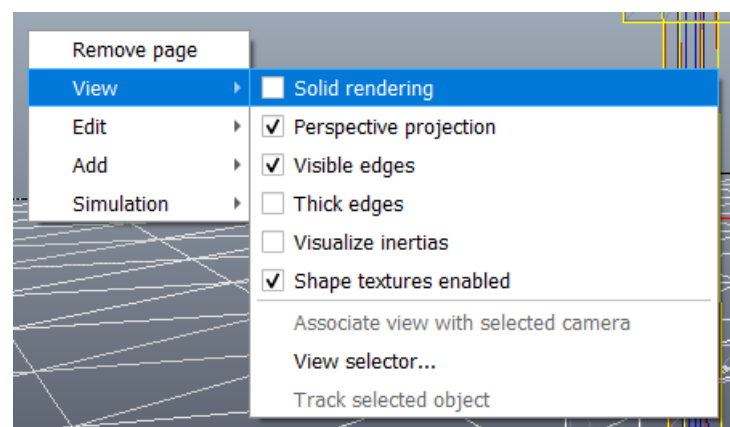
V1 — Alcune osservazioni sulle misure

Il modello finale:

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/87a85781-0e07-4db8-b111-fa6ef7dd8f3f/prismatic_v1.ttt

Visuale senza il rendering 3d

Molto utile per vedere i frame dei pezzi e poter fare una progettazione precisa.

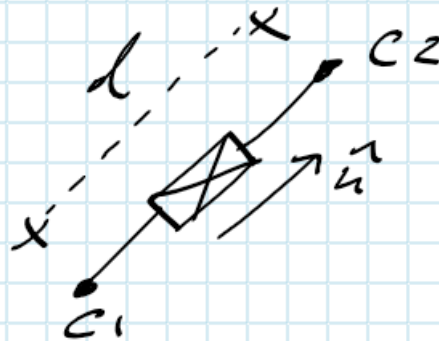


tasto destro sulla simulazione → view → toglie la spunta da Solid Rendering.

Montare correttamente un prismatic joint

Supponiamo di voler realizzare un prismatic joint con unico constraint sulla lunghezza del joint a riposo:

SCHEMATIC:

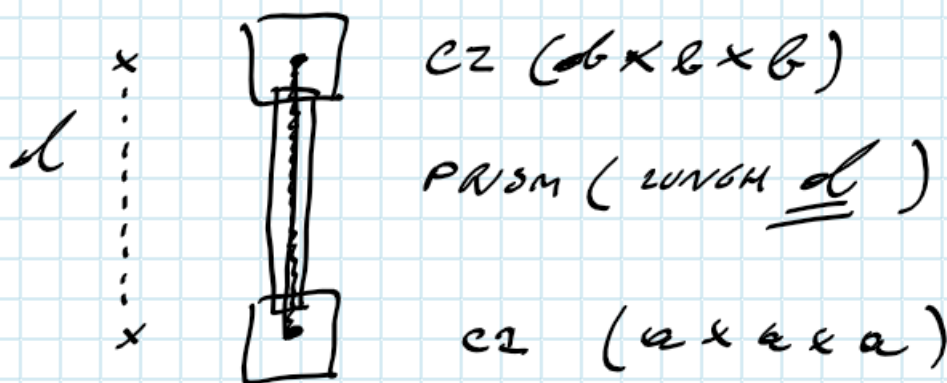


Per implementare precisamente quei constraints basta tenere conto dei centri. Supponiamo di voler realizzare il joint usando due cubi.

gerarchia:

C1
└─ PRISM
└─ C2

realizzazione in 2d: semplicemente pongo il joint lungo 'd', e 'd' la distanza tra i due cubi.

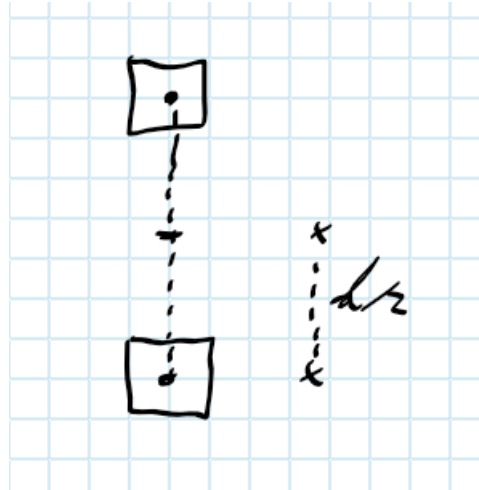


questo introduce un'eccedenza sulla misura: il joint non è lungo precisamente 'd', ma

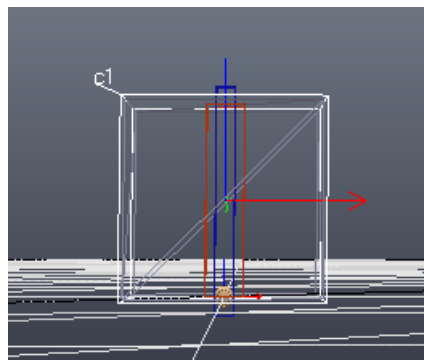
$$d + a/2 + b/2$$

Questo errore è o no accettabile? Se non stiamo lavorando con una grandissima precisione va benissimo anche così.

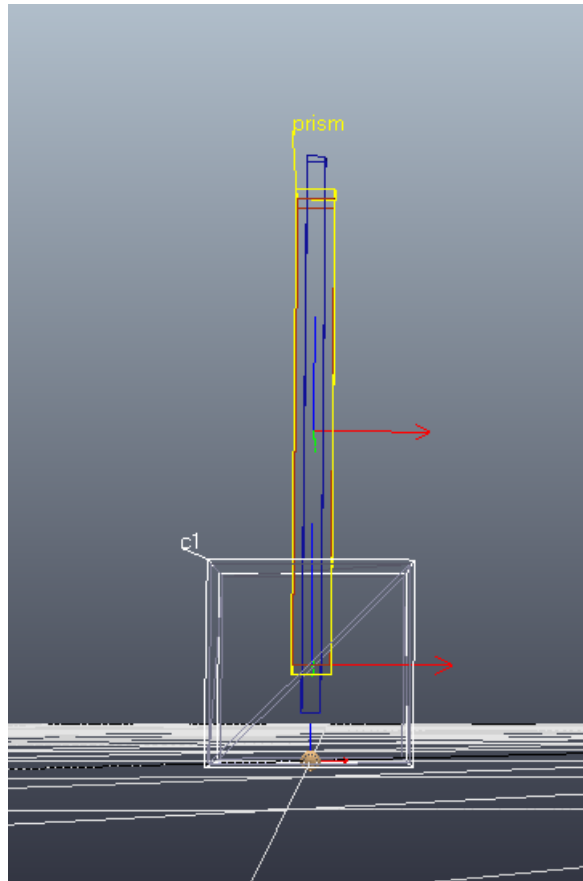
Passiamo ora alla realizzazione in CoppeliaSim. Nota che il posizionamento del joint è fatto rispetto al centro, e non rispetto ad un estremo. In termini di disegno,



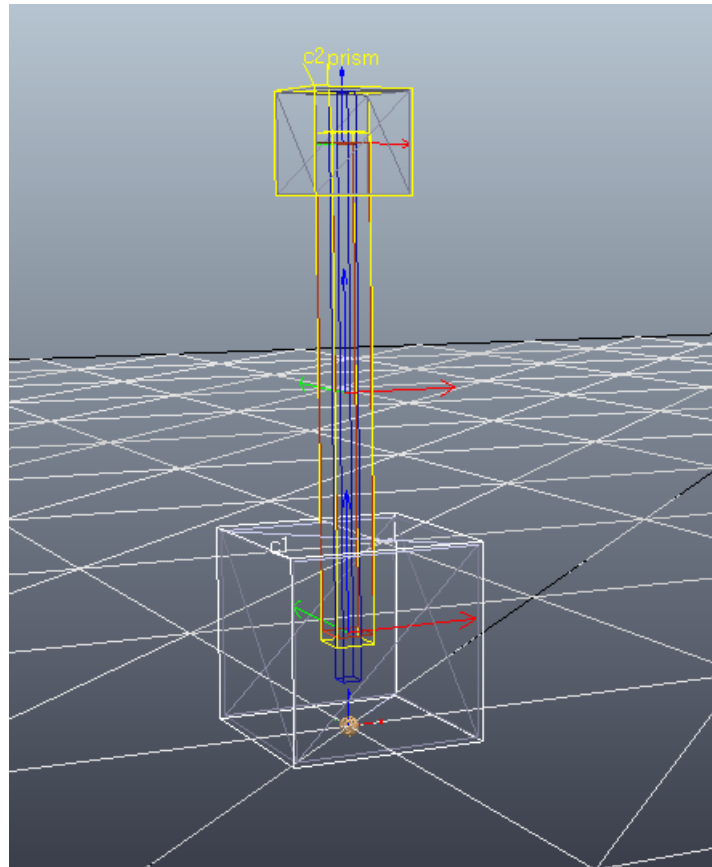
nota che l'origine di un cubo è sempre il suo centro in CoppeliaSim (esistono modi per regolare il pivot di una primitive shape????)



Il frame del cubo. Il joint è posizionato a $(0,0,0)$ rispetto al cubo, e infatti i centri coincidono, e anche la lunghezza del joint.



Il joint è lungo $d=25\text{cm}$. Posizionamento a $d/2=0.125$

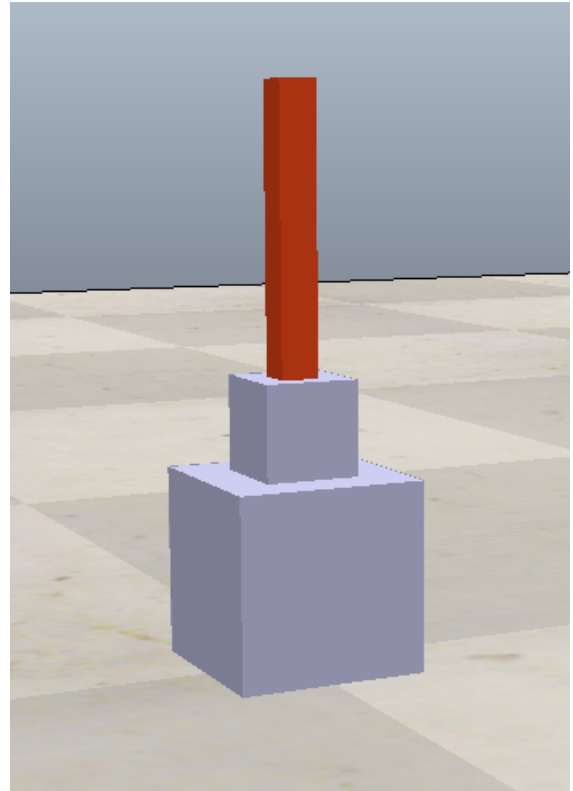
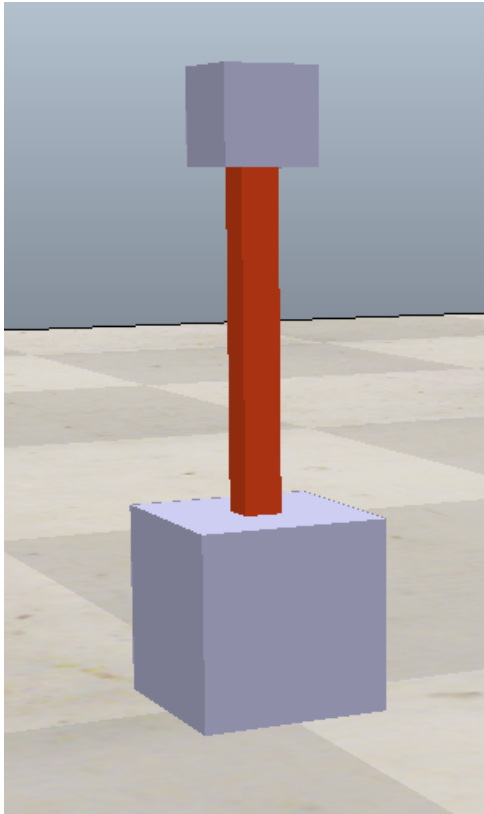


Il joint completo. Nota che va applicato lo stesso accorgimento per il posizionamento del cubo c2, che ha come padre non il cubo c1 ma il prismatic joint.

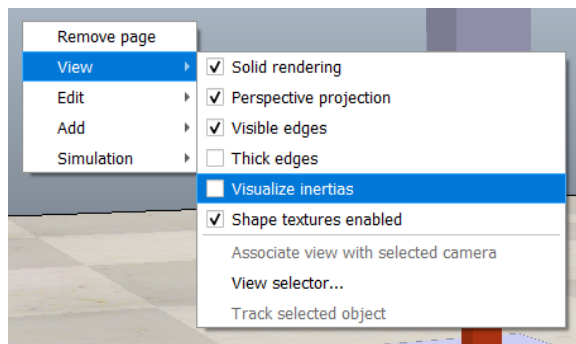
Nell'esempio: $d=0.25\text{m}$, $a=0.1\text{m}$, $b=0.05\text{m}$. Nota che, una volta montato il meccanismo, è possibile settare le dimensioni del joint senza dover riposizionare i cubi sopra e sotto: la gerarchia farà in modo che essi si riposizionino a seconda della lunghezza del joint.

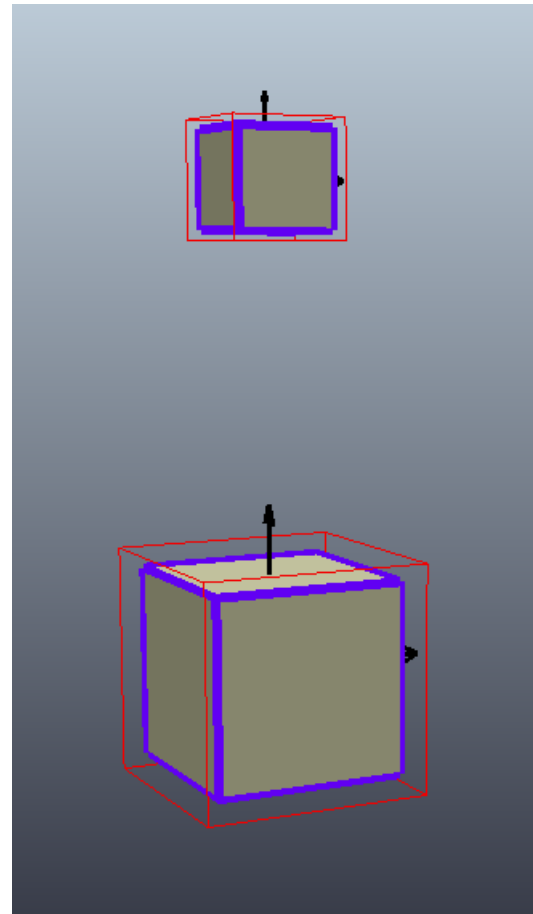
Joints e struttura

Ora che abbiamo il meccanismo, simuliamolo!



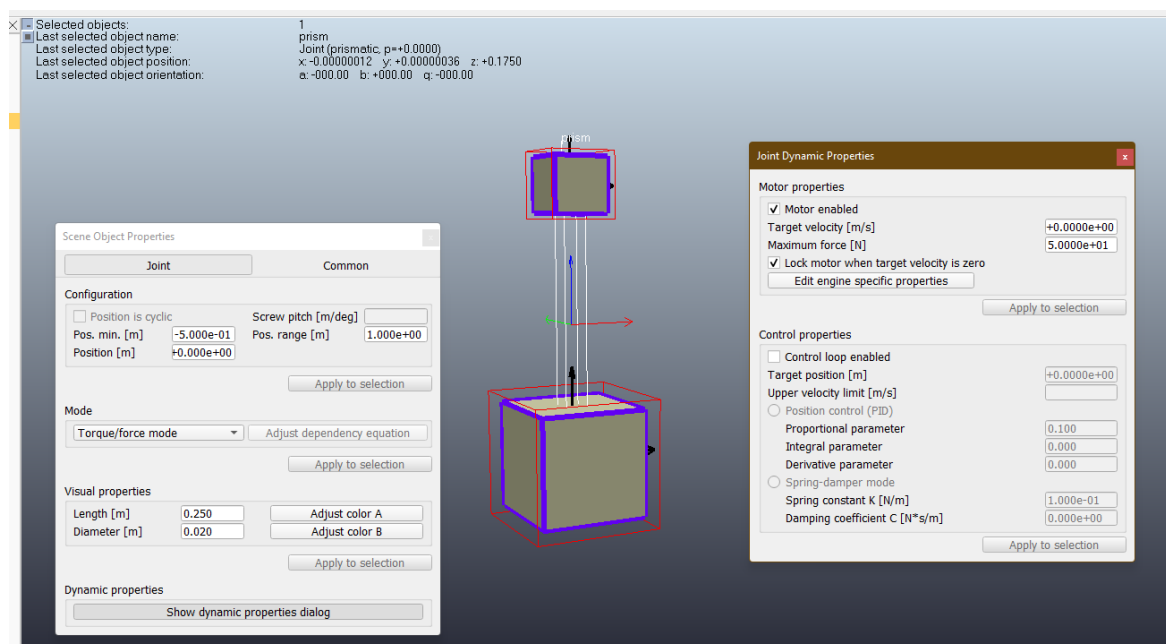
...hmmm deludente. Perché il cubo non rimane in alto? Passiamo alla visualizzazione inerziale:





Questa strana visuale indica che i due cubi sono oggetti solidi, ma non il joint! Per Coppeliasim è come se il joint non esistesse.

Il joint non è un elemento fisico, ma bensì strutturale: deve imporre un vincolo di forza interna, e basta. Per "attivare" questo vincolo, vai sulle proprietà, dynamic properties, e seleziona *motor enabled*.



Ora il cubo resta fero in aria quando attivi la simulazione, nonostante continui ad essere "sospeso nel nulla".

Regolazione del prismatic joint

Dimensioni

Visual properties		
Length [m]	<input type="text" value="0.250"/>	<input type="button" value="Adjust color A"/>
Diameter [m]	<input type="text" value="0.020"/>	<input type="button" value="Adjust color B"/>

length: la lunghezza del pistone.

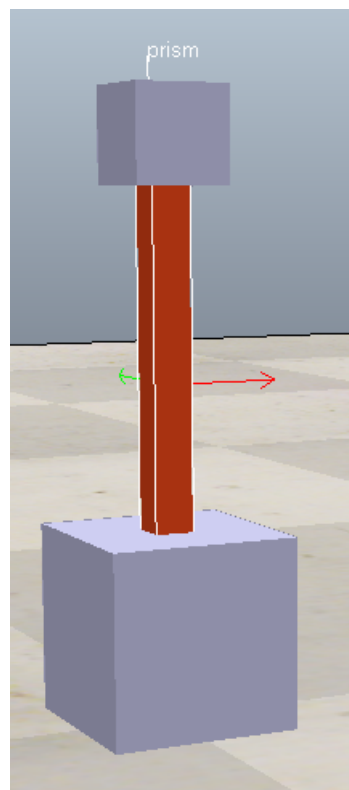
diameter: diametro del pistone (forma cubica)

Posizione e range del pistone

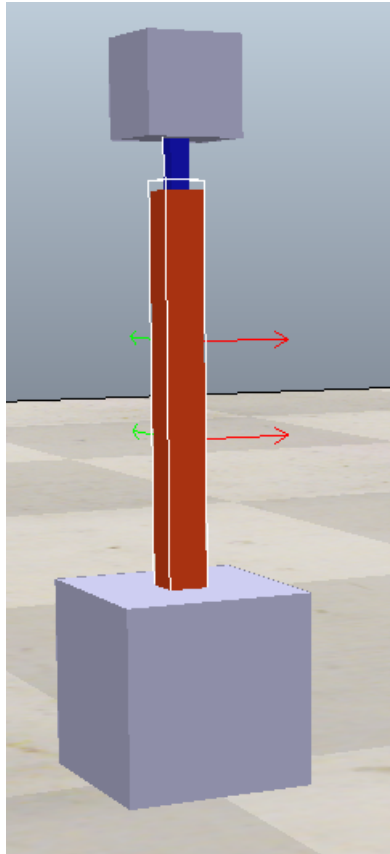
Configuration			
<input type="checkbox"/> Position is cyclic	Screw pitch [m/deg]	<input type="text" value=""/>	
Pos. min. [m]	<input type="text" value="+0.000e+00"/>	Pos. range [m]	<input type="text" value="5.000e-02"/>
Position [m]	<input type="text" value="+0.000e+00"/>		
<input type="button" value="Apply to selection"/>			

position: posizione iniziale del pistone, allungamento rispetto all'estremo opposto alla direzione del joint.

0.00



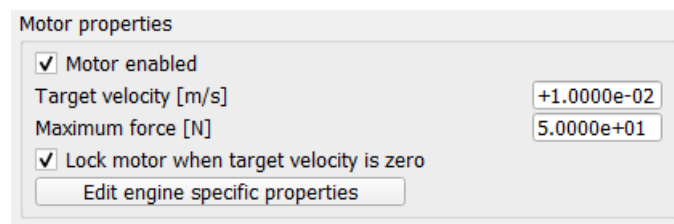
0.05



pos. min. : la minima elongazione del pistone.

pos. range: la massima elongazione del pistone.

Proprietà dinamiche



target velocity (disattivata in modalità *control loop enabled*) velocità del pistone, in metri al secondo.

maximum force: la massima spinta che il pistone può esercitare nella sua direzione. Nota che questa forza è concorde con tutto quello che sai di fisica: se ad esempio il punto su cui preme il pistone fosse troppo pesante, potrebbe generarsi un movimento oscillatorio stile molla, oppure creare una forza opposta alla direzione del pistone, oppure un corpo troppo pesante potrebbe bloccare il pistone.

Motor properties

☒ Motor enabled

Target velocity [m/s]

Maximum force [N]

☐ Lock motor when target velocity is zero

Control properties

☒ Control loop enabled

Target position [m]

Upper velocity limit [m/s]

☒ Position control (PID)

Proportional parameter

Integral parameter

Derivative parameter

☐ Spring-damper mode

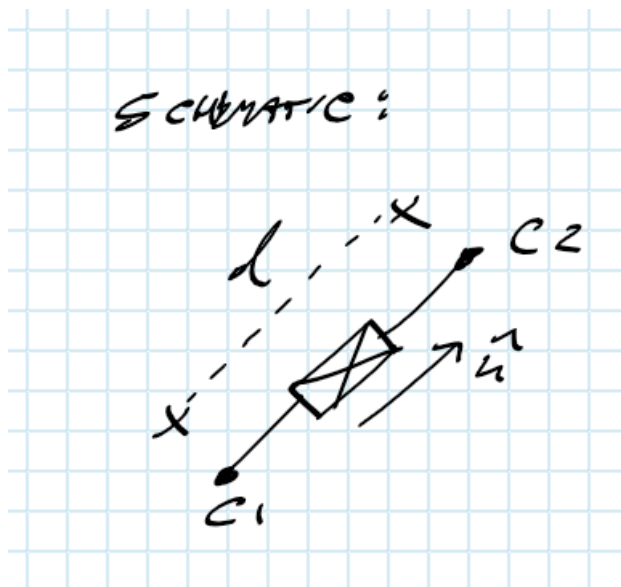
Spring constant K [N/m]

Damping coefficient C [N*s/m]

In modalità *control loop enabled* il dispositivo può essere anche controllato da codice. Le funzioni di coppeliaSim implementano un controllore PID sul pezzo, già pronto per l'uso.

V2 — prismatic joint bello

In questo modello, creo e uso qualche mesh. Il meccanismo è lo stesso presentato prima: il solito prismatic che lega i due punti C1 e C2.



Per l'implementazione però stavolta voglio usare dei cilindri, e renderlo leggermente più appetibile all'occhio.

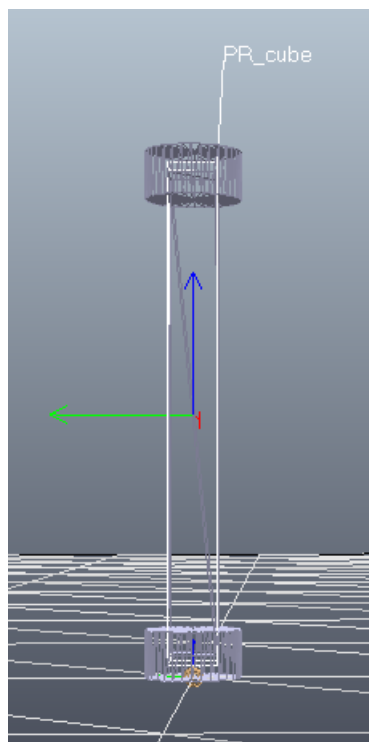
Prima impostazione del joint

Misure:

- c1, c2:
 - diametro: 0.05m
 - lunghezza: 0.025m
- PR_cube
 - d: 0.25m
 - lato: 0.025m

Attenzione: quando usi un cubo al posto del prismatic come fatto prima, non puoi permetterti l'errore di prima sulla lunghezza, perchè altrimenti si verificano strane forze interne.

Nel seguente, nota che il pezzo entra dentro il cilindro superiore, e anche dentro il cilindro inferiore.



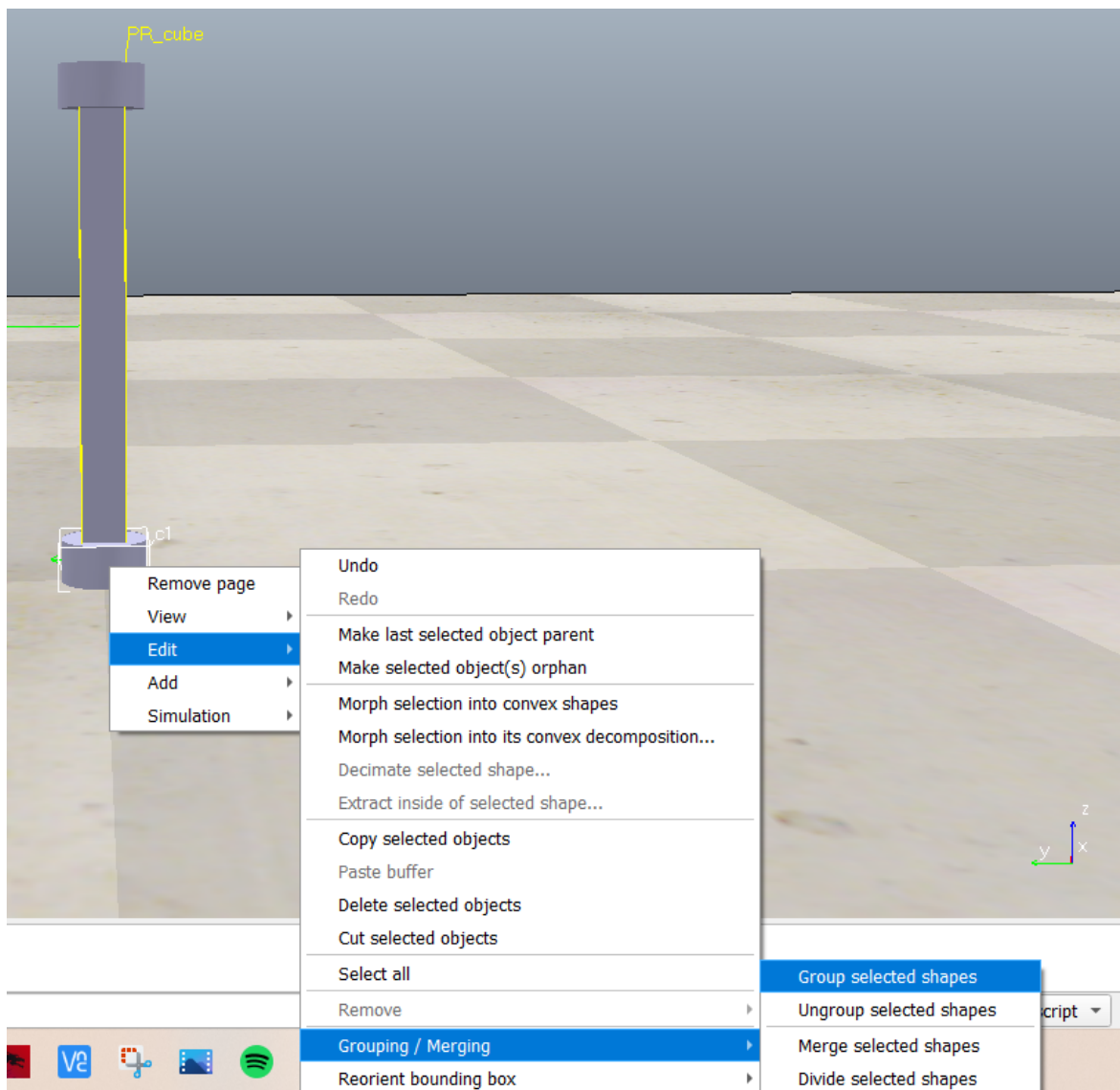
Se fai partire così la simulazione, il pezzo si storce tutto e può anche cadere. Fare grouping con queste misure non risolve il problema: anche col grouping si hanno delle forze interne che causano il moto accidentale del pezzo.

La corretta posizione del pezzo centrale non è più semplicemente $d/2$, ma

$$d/2 + 0.0125 \rightarrow 0.125 + 0.0125 \rightarrow \mathbf{0.1375}$$

Fatto questo aggiustamento per entrambi i cilindri, il dispositivo vibra un pochino in simulazione, ma nulla di preoccupante. Sicuramente non casca, o non va a finire nella stratosfera.

Grouping dei due pezzi inferiori:

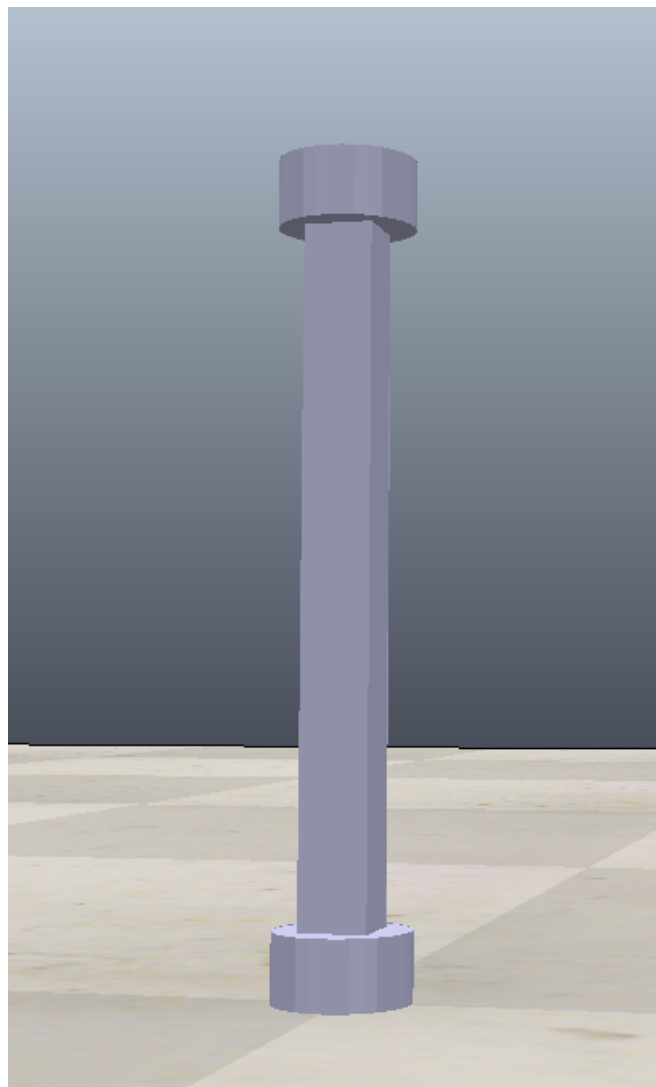
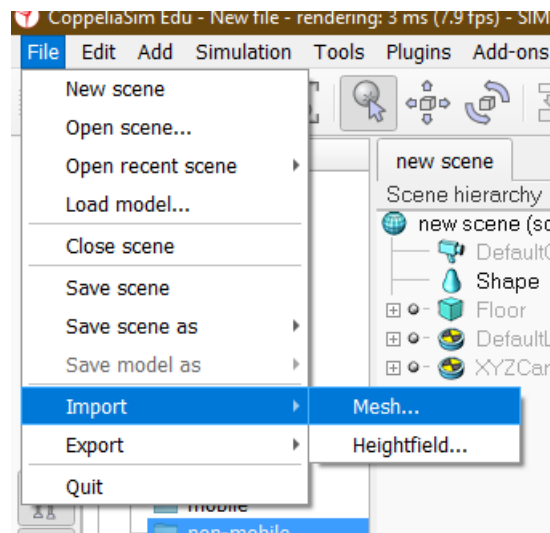


Ma siccome voglio esportare la mesh, posso direttamente fare grouping su tutti i pezzi.

Lavorare su una mesh

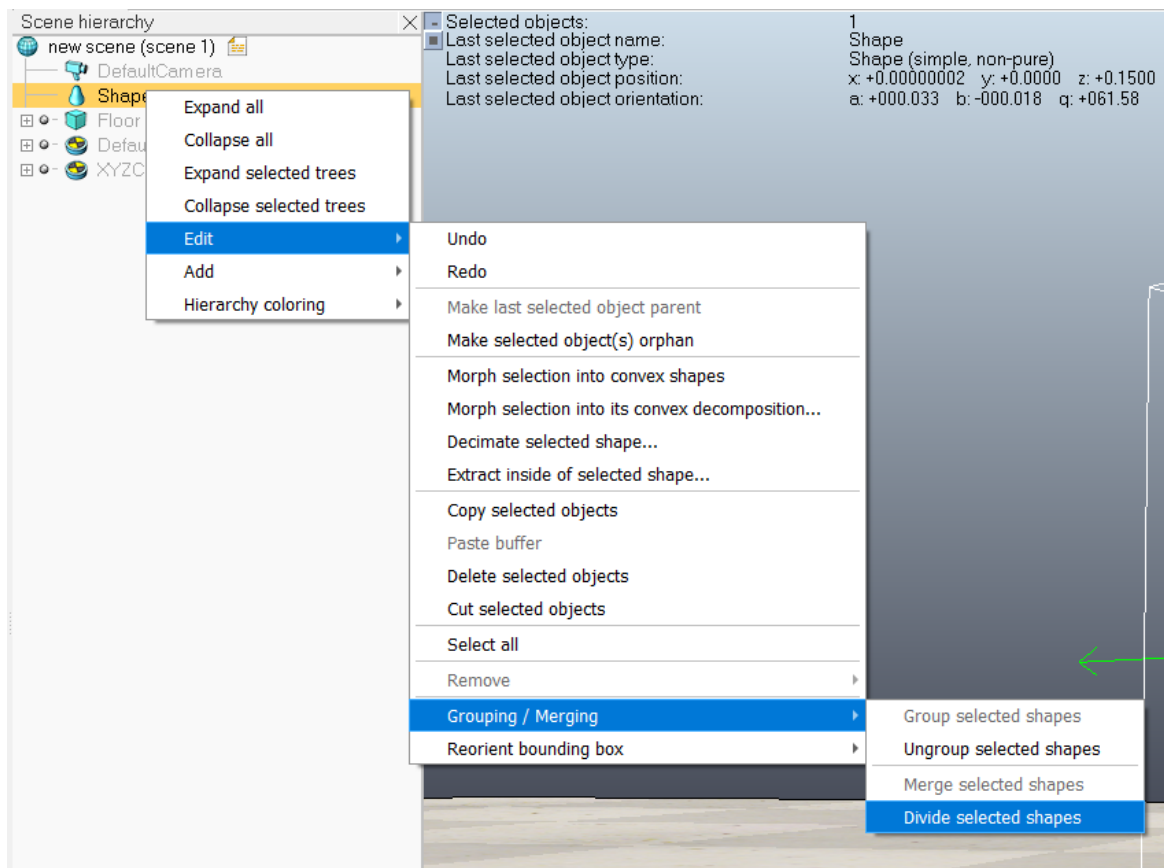
Usiamo il prismatic come dovrebbe essere usato, vale a dire *come vincolo strutturale* e non come parte integrante della struttura.

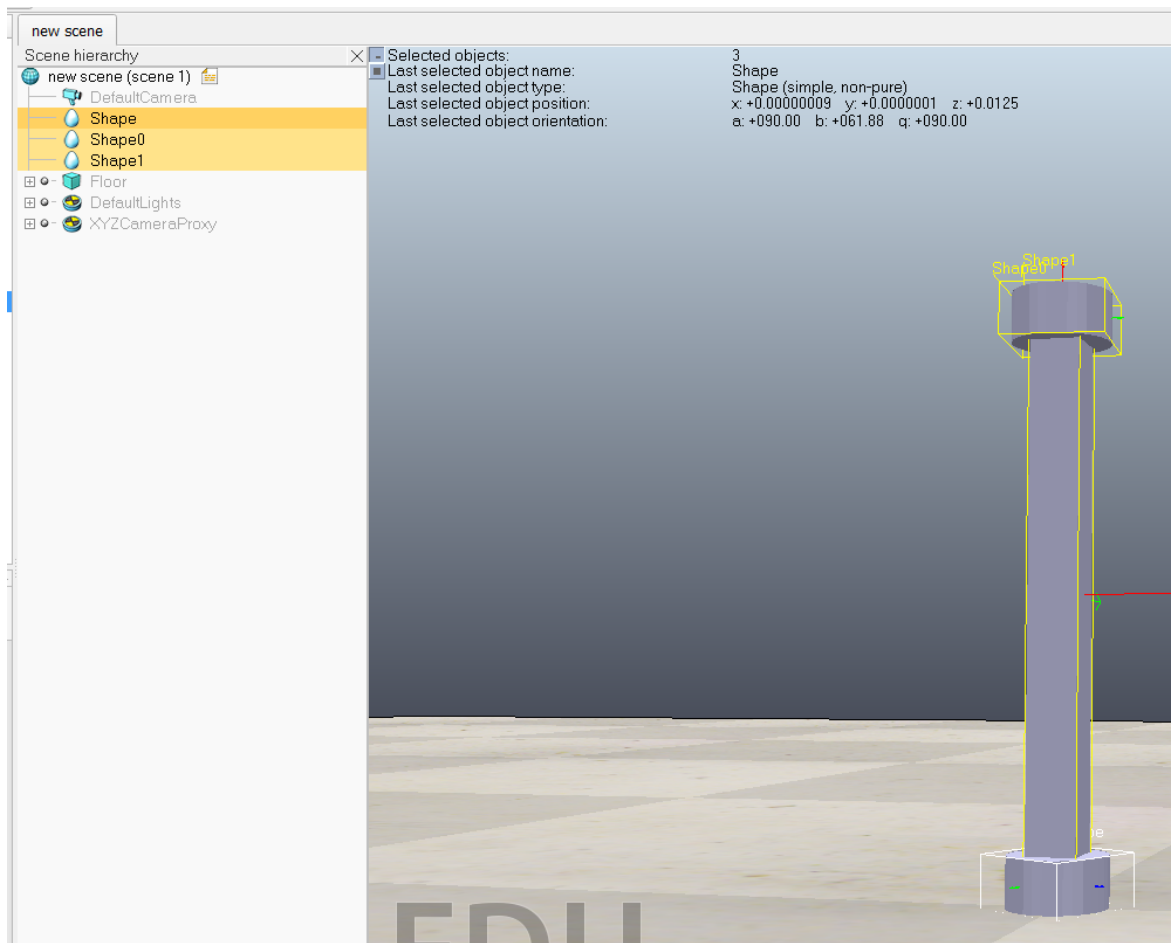
Importazione della mesh:



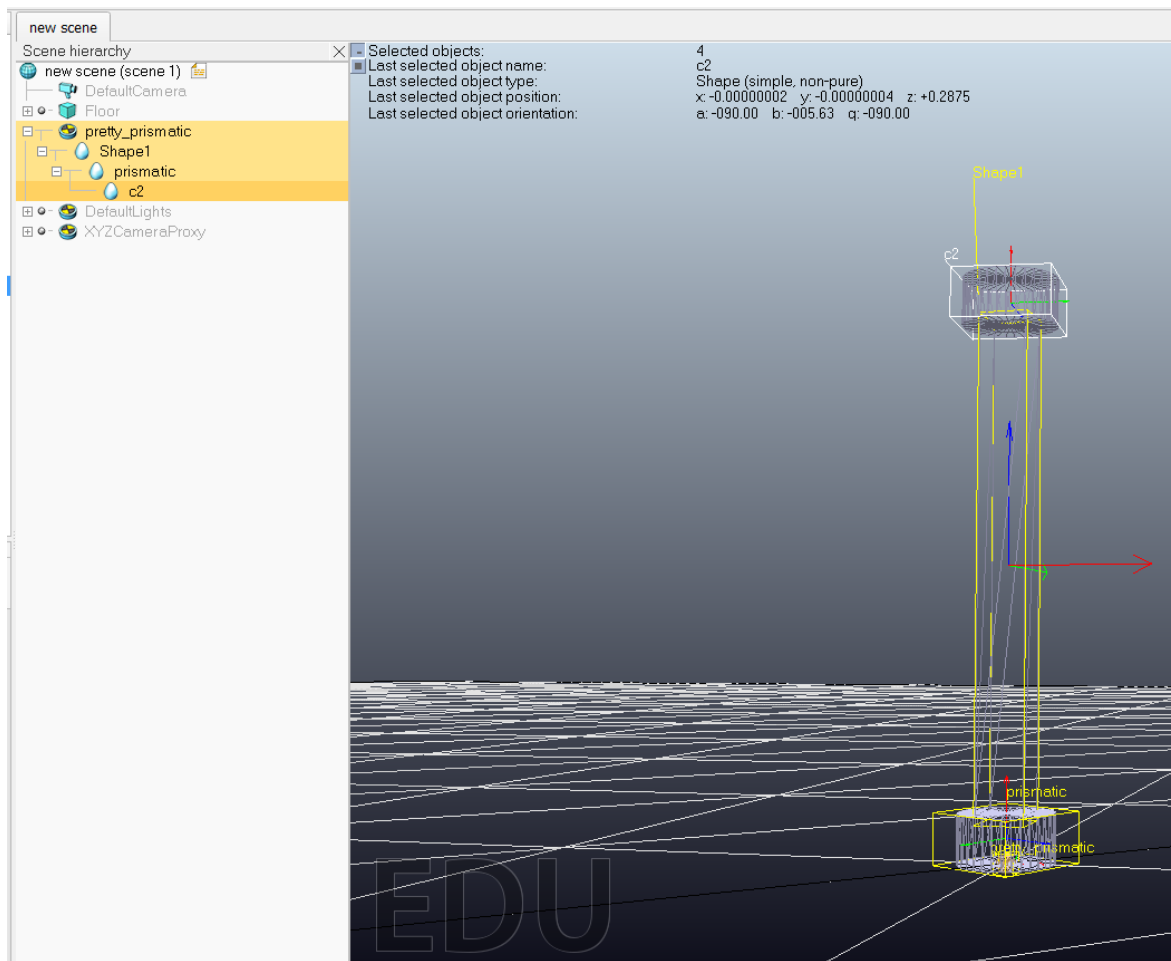
Nota che la mesh ha perso leggermente qualità grafica: questo perchè in effetti nello step precedente non era stata data alcuna impostazione grafica...

La mesh è vista come un oggetto unico, ma possiamo chiedere a CoppeliaSim di suddividere le shapes.



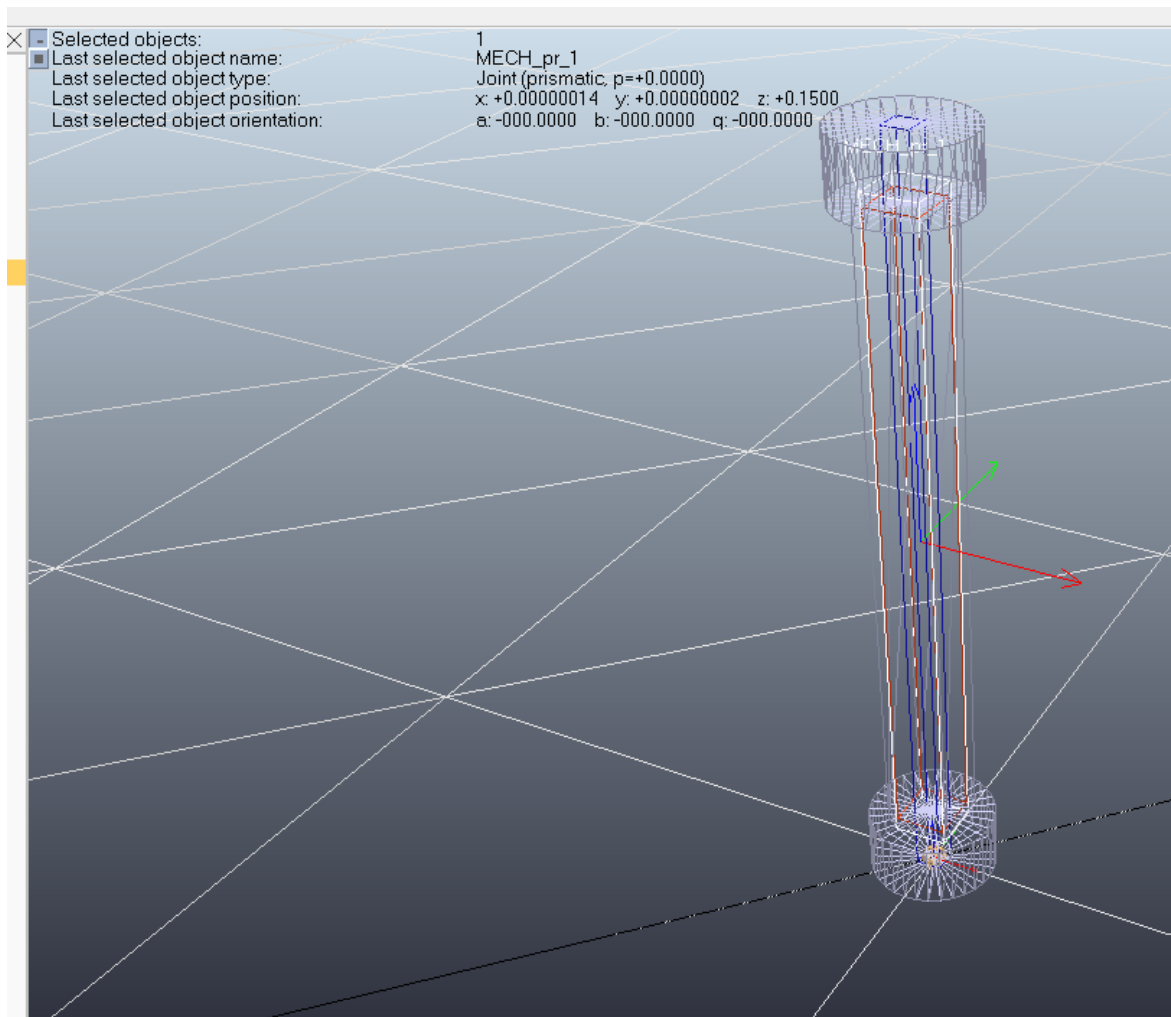


ecco come appare la suddivisione delle mesh



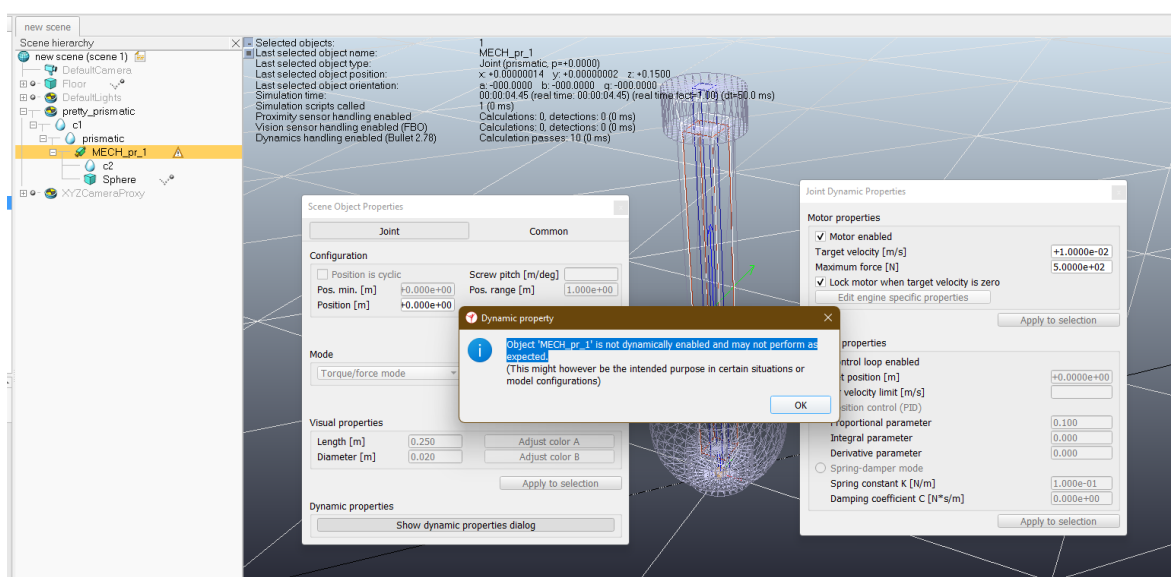
Nota che la mesh non ha ancora impostazioni fisiche, quindi se provi ad avviare la simulazione vedrai che il componente non ha alcuna oscillazione.

Mettiamo il pistone. Attenzione: dev'essere posizionato all'origine dell'involucro del prismatic.



Problemi di fisica...

Se provi ad avviare la simulazione così com'è... succede questo:



...ignora la sfera...

Messaggio d'errore:


Object 'MECH_pr_1' is not dynamically enabled and may not perform as expected.

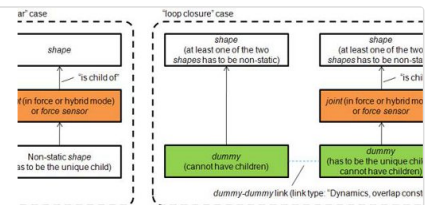
Maggiori informazioni qui:

Designing dynamic simulations

In CoppeliaSim, only a limited number of objects will be dynamically simulated.

Those are shapes, joints and force sensors, but it will depend on the scene structure and object properties, whether a given object will be dynamically

 <https://www.coppeliarobotics.com/helpFiles/en/designingDynamicSimulations.htm>



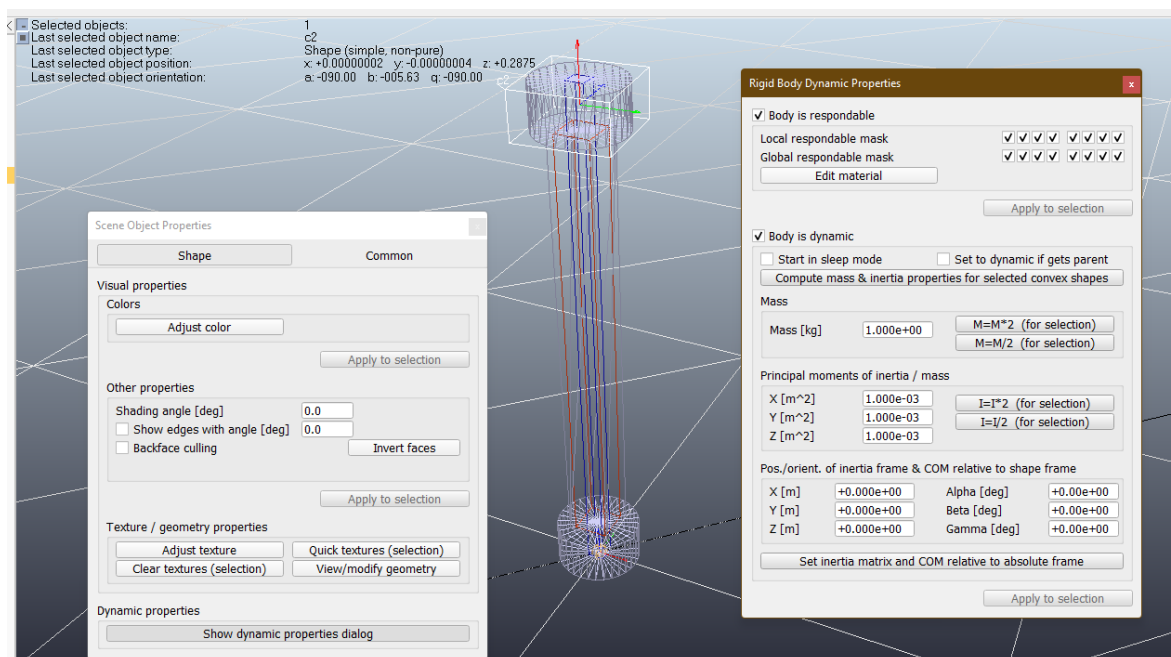
e "utilissima" risposta sul forum:

Forum

Hello! I would like to know how to become object IRB4600_joint1 dynamically enable. Thank you! Hello, have a look here. But the robot model you are referring to is entirely static, since it is anyway not supposed to collide with the environment.

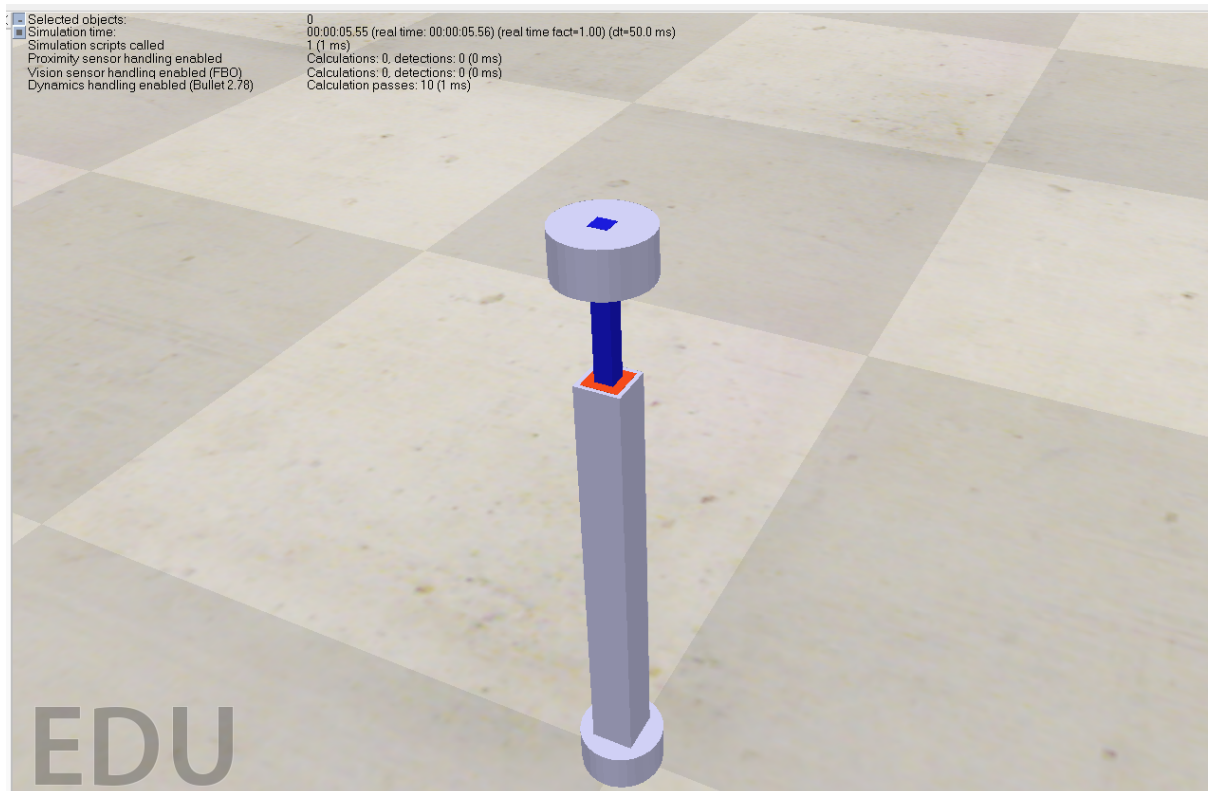
 <https://forum.coppeliarobotics.com/viewtopic.php?t=7056>

Per risolvere il problema, ricorda di rendere dinamico l'oggetto da muovere!



Fatto questo, tutto dovrebbe funzionare come ci si aspetta.

Il risultato



scene con i cubi:

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/18c85778-f659-48ec-8e17-3272318da172/prismatic_v2.ttt

scene finale col joint e la mesh:

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/742772bb-27b9-4a9a-b4bd-8a66e73f2083/prismatic_v2_mesh.ttt

mesh del pistone:

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/c8166eef-a7d5-45f8-a805-ebd3e57c0d11/prismatic_CAD_model.mtl

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/b0ce2929-563d-411b-aa21-abdef77d2594/prismatic_CAD_model.obj