Les scènes mécaniques

J. Bayle, J. M. Tisserand, A. Tran Van

Résumé : On propose de réaliser un programme, basé sur les primitives graphiques de Mathematica, permettant de construire le schéma cinématique d'un mécanisme donné.
Mots-clés : mécanique, liaisons cinématiques, graphe des liaisons, volumes élémentaires.

Abstract : We carry out a program which allows to produce a graphical representation of a linkage.
Keywords : mechanics, kinematic links, graph of links, elementary bulks.

Remarque : Pour exécuter le programme, il faut tout d'abord évaluer l'accolade principale de la partie annexe.

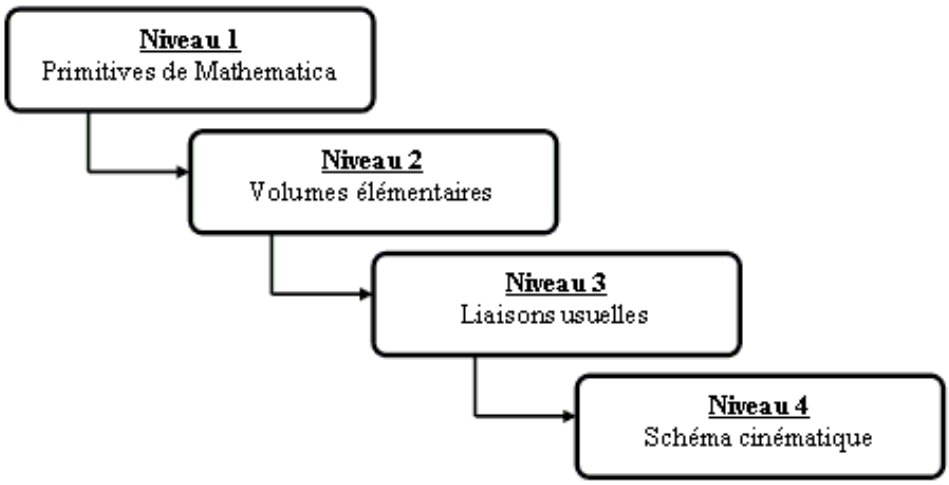
Introduction

Ce projet a pour but d'élaborer un programme permettant de construire un schéma cinématique. Pour cela, nous partons d'un mécanisme donné, ou plus exactement d'un graphe des liaisons symbolisant les composants de ce mécanisme. Nous avons choisi ce sujet pour son aspect concret et graphique qui permet de visualiser les résultats au fur et à mesure de l'avancement.
Dans un premier temps, nous allons répertorier les liaisons cinématiques usuelles principales ainsi que les outils nécessaires à leurs réalisations. Une fois ces outils définis, nous avons créer un programme permettant l'assemblage des liaisons. Par la suite, nous reprendrons des lignes de ce programme afin d'expliquer notre démarche de programmation. Pour finir, nous illustrerons l'usage de ce programme par un exemple.

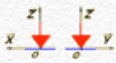
Etude préliminaire

Avant de se lancer dans le cœur du projet, nous avons décidé de nous pencher sur les principales étapes de l'étude, c'est-à-dire :
- réalisation des liaisons cinématiques
- présentation du graphe des liaisons
- construction du schéma cinématique
Nous avons choisi d'établir le schéma cinématique en trois dimensions afin que son interprétation soit abordable par tous.

Notre raisonnement peut donc être hiérarchisé de la façon suivante :



On peut voir ci-dessous les représentations usuelles en deux et trois dimensions de quelques liaisons cinématiques.

Nom de la liaison	Représentations planes	Perspective	Degrés de liberté	mobilités
Liaison ***glissière*** *de centre A et d'axe X*			Translation Rotation **Tx** 0 0 0 0 0	
Liaison ***pivot*** *de centre A et d'axe X*			Translation Rotation 0 **Rx** 0 0 0 0	
Liaison ***Pivot Glissant*** *de centre C et d'axe X*			Translation Rotation **Tx** **Rx** 0 0 0 0	
Liaison ***Appui Plan*** *de centre D et de normale Z*			Translation Rotation **Tx** 0 **Ty** 0 0 **Rz**	
Liaison ***rotule*** *de centre O*			Translation Rotation 0 **Rx** 0 **Ry** 0 **Rz**	
Liaison ***linéaire annulaire*** *de centre B et d'axe X*			Translation Rotation **Tx** **Rx** 0 **Ry** 0 **Rz**	
Liaison ***linéïque rectiligne*** *de centre C, d'axe X et de normale Z*			Translation Rotation **Tx** **Rx** **Ty** 0 0 **Rz**	
Liaison ***ponctuelle*** *de centre O et de normale Z*			Translation Rotation **Tx** **Rx** **Ty** **Ry** 0 **Rz**	

■ Les liaisons cinématiques

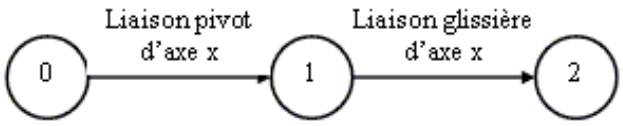
Il faut tout d'abord répertorier les liaisons cinématiques usuelles susceptibles d'intervenir dans un mécanisme. Par la suite, nous limiterons l'usage de ce programme à des mécanismes simples en boucle ouverte. Nous avons décidé de ne pas représenter la liaison hélicoïdale, par souci de simplicité, ainsi que la rotule à doigt car peu utilisée. Au final, nous aurons donc huit liaisons.
Chaque liaison cinématique est représentée à partir de volumes élémentaires tels que des sphères, des plans, des prismes...
Certains volumes sont issus de la bibliothèque : `Graphics`Shapes`, d'autres ont été créés par nos soins.
Nous avons donc créé une fonction pour chaque liaison permettant alors de les afficher.

1) Liaison glissière : 1 pavé plein, 1 pavé creux.
2) Liaison pivot : 1 cylindre plein, 1 cylindre creux .
3) Laison pivot glissant : 1 cylindre plein, 1 cylindre creux, 2 disques.
4) Liaison appui plan : 2 plans.
5) Liaison rotule : 1 sphère coupée, 1 sphère pleine.
6) Liaison linéaire annulaire : 1 cylindre coupé, 1 sphère pleine.
7) Liaison linéaire rectiligne : 1 prisme droit, 1 plan.
8) Liaison ponctuelle : 1 cône, 1 plan.

On pourra trouver dans le programme, en annexe, le détail des différentes fonctions ainsi que des illustrations pour comprendre leur fonctionnement.

■ Le graphe des liaisons

Ce graphe fait le lien entre le mécanisme réel donné et le schéma cinématique. Ses sommets représentent les différents blocs cinématiques, tandis que les connexions entre les sommets identifient les liaisons inter-blocs. Ces informations seront fournies à partir d'une structure de données de type texte, et non à partir d'un graphique. Sur l'exemple fictif suivant, nous pouvons voir les deux modèles de graphes énoncés précédemment. Le premier est représenté graphiquement et le second sous forme de données.



Dans notre cas,nous prendrons ce modèle de graphe des liaisons:
0 : bloc A
1 : bloc B
2 : bloc C
L 0-1 : pivot d'axe x
L 1-2 : glissière d'axe x

Voici la forme qu'aurait le graphe des liaisons avec Mathematica :

```
ms = MechanicalSystem[{
Vertex[s0, Body[m0, {1, 0, 1}]],
Vertex[s1, Body[m1, {0, 1, 0}]],
Joint[{s0, s1}, Pivot[{0, 0, 0}, {0, 0, 0}]],
Vertex[s2, Body[m2, {1, 0, 0}]],
Joint[{s1, s2}, Glissiere[{0, 8, 8}, {0, 0, 0}]]
}]
```

Le schéma cinématique

L'association des liaisons cinématiques avec le graphe des liaisons permet d'aboutir au schéma cinématique. Chaque élément des solides formant un bloc cinématique sera affecté d'une même couleur. Le problème réside en l'assemblage cohérent des différentes liaisons. Il faudra prendre soins de respecter l'ordre des liaisons.

Conception du programme

Nous trouverons ci-dessous un exemple commenté de fonction, de liaison et de schéma cinématique. L'ensemble se trouve en annexe.

Fabrication des fonctions

Pour chaque volume, nous avons créé une fonction ayant une structure quasi-similaire à celle ci-dessous :

```
FormePave[{x1_,y1_,z1_},{x2_,y2_,z2_},Col_,Thic_]:={
```

On nomme la fonction (elles commenceront toutes par Forme... suivi du nom l'identifiant). On définit ensuite ses variables : les coordonnées des deux points diagonalement opposés, la couleur et l'épaisseur du trait.

```
EdgeForm[{GrayLevel[0],Thickness[Thic]}],
```

Définition des caractéristiques de l'arête : couleur et épaisseur.

```
SurfaceColor[Apply[RGBColor,Col]],
```

Définition des paramétres de la couleur de la surface.

```
Polygon[{{x1,y1,z1},{x2,y1,z1},{x2,y2,z1},{x1,y2,z1}}],
Polygon[{{x1,y1,z1},{x2,y1,z1},{x2,y1,z2},{x1,y1,z2}}],
Polygon[{{x2,y1,z1},{x2,y2,z1},{x2,y2,z2},{x2,y1,z2}}],
Polygon[{{x2,y2,z1},{x1,y2,z1},{x1,y2,z2},{x2,y2,z2}}],
Polygon[{{x1,y2,z1},{x1,y1,z1},{x1,y1,z2},{x1,y2,z2}}],
Polygon[{{x1,y1,z2},{x2,y1,z2},{x2,y2,z2},{x1,y2,z2}}]};
```

Représentation des surfaces formant le pavé.

Maintenant, nous avons à notre disposition tous les outils nécessaires pour afficher les liaisons.

■ Fabrication des liaisons

L'exemple qui suit permet de voir comment intervient une fonction au sein d'une liaison.

GraphicForm[
Repésentation d'un objet auquel on affecte des paramètres : nom de la liaison et couleur des deux solides la constituant.
Rectiligne[t_,{rα_,rβ_,rγ_}],Col1_,Col2_]:=
Nom de la liaison avec ses paramètres de translation et de rotation la positionnant dans l'espace.
TranslateShape[
Permet de translater l'image suivant les axes X, Y, Z.
RotateShape[{
Permet d'orienter l'image suivant les angles d'Euler
FormePrisme[{0,0,0},{-1.5,2,5},Col1,0.001],
Représentation d'un prisme.

```
RotateShape[
FormePlan[{-3,-1,0},{3,6,0},Col2,0.001],0,-π/2,0],
FormeSegment[{0,0,2.5},{0,-5,2.5},Col2,0.01],
FormeSegment[{0,2,2.5},{0,5,2.5},Col1,0.01]},rα,rβ,rγ],t];
```

Nous trouverons la repésentation graphique de cette fonction en annexe. Il ne nous reste plus qu'à créer les fonctions qui assembleront les liaisons.

■ Fabrication du schéma cinématique

Nous avons créé plusieurs fonctions, détaillées ci-dessous, destinées à manipuler les liaisons et nous permettant de construire le schéma cinématique.

Fonction imposant les couleurs à une liaison.

```
Couleur[r_,s_]:= Part[Cases[s,Vertex[r,Body[m_,c_]]→c],1];
```

Fonction construisant la liaison.

```
Auxiliary[Joint[{s1_,s2_},link_],s_]:=
GraphicForm[link,Couleur[s1,s],Couleur[s2,s]];
```

Fonction traçant le schéma cinématique. Le point de vue de la représentation pourra être modifié par l'utilisateur en changeant les paramétres de **ViewPoint.**

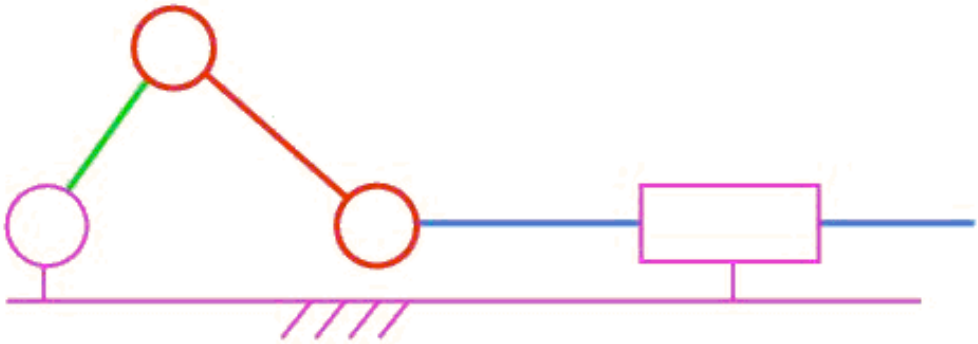
```
GraphicForm[MechanicalSystem[s_],opt___]:=
Show[Graphics3D[Map[Auxiliary[#,s]&,Cases[s,_Joint]]],opt,
PlotRange→All,Boxed->False,ViewPoint →{2,1,2},ImageSize→{400,400},
PlotRange->All,PlotLabel->"MechanicalSystem";
```

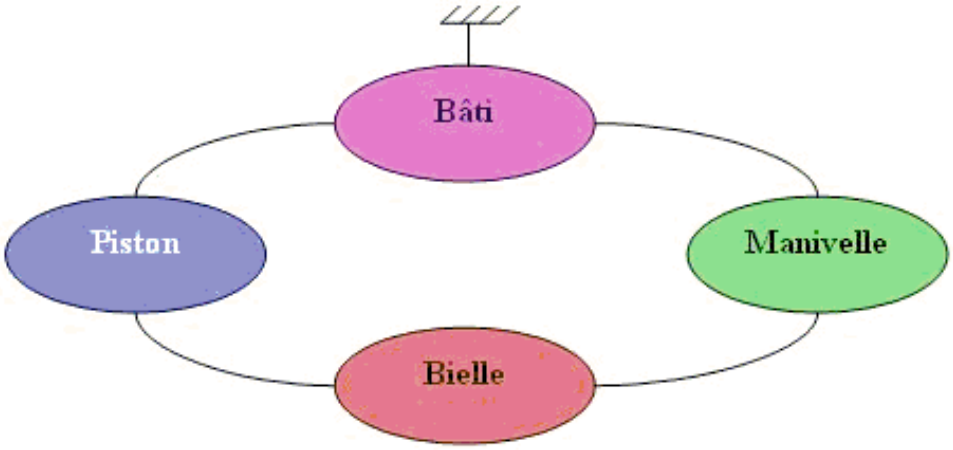
Nous effectuons ensuite une application mettant en œuvre le programme.

Application

Nous avons traité plusieurs exemples afin de valider les fonctions mises en jeu. Ces systèmes sont consultables en annexe.

Nous avons choisi un mécanisme concret en boucle fermé : "le système bielle-manivelle". Nous trouverons ci-dessous le schéma cinématique en deux dimensions et le graphe des liaisons (ces deux éléments n'ont pas été réalisés avec *Mathematica*) afin de faire le lien entre la représentation habituelle et celle de notre programme.

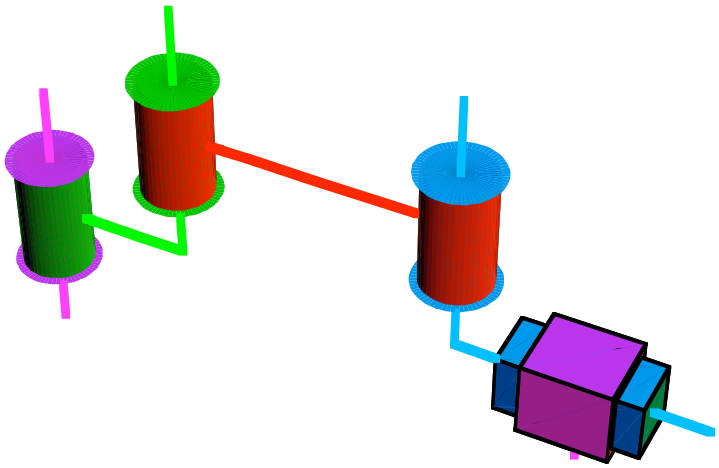




L'extrait de programme suivant permet d'obtenir le schéma cinématique en trois dimensions du système précédent.

```
ms = GraphicForm[MechanicalSystem[{
     Vertex[s0, Body[m0, {1, 0, 1}]],
     Vertex[s1, Body[m1, {0, 1, 0}]],
     Joint[{s0, s1}, Pivot[{0, 0, 0}, {π, 0, 0}]],
     Vertex[s2, Body[m2, {1, 0, 0}]],
     Joint[{s1, s2}, Pivot[{0, 8, 8}, {π, 0, 0}]],
     Vertex[s3, Body[m3, {0, 0.7, 1}]],
     Joint[{s3, s2}, Pivot[{0, 24, 8}, {0, 0, 0}]],
     Vertex[s4, Body[m4, {1, 0, 1}]],
     Joint[{s3, s4}, Glissiere[{0, 31, 0}, {0, π / 2, 0}]]
    }]]
```

MechanicalSystem



- Graphics3D -

L'obtention du résultat confirme le bon fonctionnement du programme. L'assemblage est cohérent et symbolise bien le systéme réel.

▪ Discussion et perspectives

▪ Discussion

Un schéma cinématique peut se représenter aussi bien en deux dimensions qu'en trois dimensions. Nous avons choisi de travailler en trois dimensions afin de faciliter le travail sur *Mathematica*. De plus, une représentation en perspective rend le fonctionnement du mécanisme plus facile à comprendre surtout si l'on veut qu'il soit abordable par des novices. Nous avons tenté d'écrire les expressions du programme de façon simple et réduite ainsi que d'introduire l'usage des couleurs pour identifier plus facilement les différentes parties d''un mécanisme.

Perspectives

Par la suite, on pourra faire une programmation permettant de visualiser les mouvements d'un mécanisme à l'aide des commandes d'animation de *Mathematica*. Enfin dans la mesure du possible, on tentera de simplifier les expressions du programme. Dans un cadre plus approfondi, la réalisation d'un schéma cinématique réduit pourrait être étudiée c'est-à-dire obtenir un schéma cinématique représentant le mécanisme étudié mais avec le minimum de liaisons possibles.

Conclusion

Ce programme permet d'obtenir, à partir d'un mécanisme, le schéma symbolique de fonctionnement en trois dimensions. Nous avons recontré le plus de difficultés pour assembler les liaisons de façon cohérente. Cependant nous sommes contents du résultat obtenu au vu des ambitions du projet. Voir que notre programme a abouti à quelque chose de concret nous apporte une grande satisfaction personnelle. Nous avons pu juger de la puissance de *Mathematica* ainsi que des nombreux avantages (temps de programmation reduit, bibliothéque plus fournie....) qu'il apporte face à des logiciels dont le langage est moins évolué.

Bibliographie

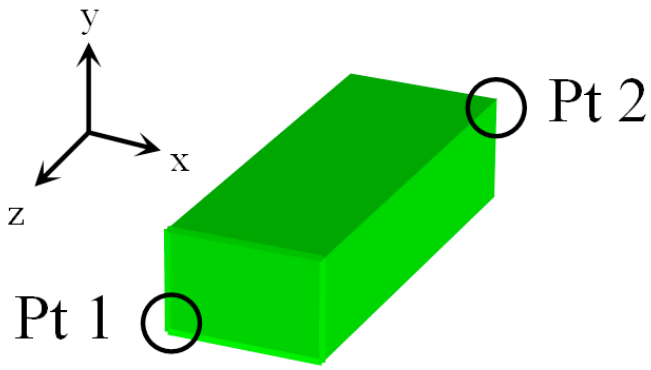
[1] TROTT Mickael The *Mathematica* guidebook for graphics, 2004 - Springer
[2] WHICKAM JONES Tom *Mathematica* graphics technics and applications, 1994 - Telos
[3] GRAY Alfred Modern differential geometry of curves and surfaces, 1993 - CRC Press
[4] SKIENA Steven Combinatorics and graphics theory with *Mathematica,* 1990 - Addison Wesley Publishing Co.
[5] S. CLATOT, H. FAVRELIERE, A. LE LOUARN, T. VERON Modélisation des mécanismes, rapport de projet 2003 ENSMM
[6] Site internet : http://barreau.matthieu.free.fr

Annexes

```
<< Graphics`Shapes`
```

Fonction réalisant un pavé plein avec les coordonnées de 2 sommets opposés

```
FormePave[{x1_, y1_, z1_}, {x2_, y2_, z2_}, Col_, Thic_] := {
   EdgeForm[{GrayLevel[0], Thickness[Thic]}],
   SurfaceColor[Apply[RGBColor, Col]],
   Polygon[{{x1, y1, z1}, {x2, y1, z1}, {x2, y2, z1}, {x1, y2, z1}}],
   Polygon[{{x1, y1, z1}, {x2, y1, z1}, {x2, y1, z2}, {x1, y1, z2}}],
   Polygon[{{x2, y1, z1}, {x2, y2, z1}, {x2, y2, z2}, {x2, y1, z2}}],
   Polygon[{{x2, y2, z1}, {x1, y2, z1}, {x1, y2, z2}, {x2, y2, z2}}],
   Polygon[{{x1, y2, z1}, {x1, y1, z1}, {x1, y1, z2}, {x1, y2, z2}}],
   Polygon[{{x1, y1, z2}, {x2, y1, z2}, {x2, y2, z2}, {x1, y2, z2}}]};
```

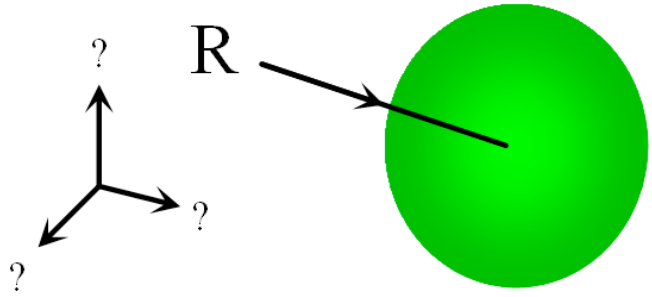


Fonction réalisant un pavé filaire avec les coordonnées de 2 sommets opposés

```
FormePaveFilaire[{x1_, y1_, z1_}, {x2_, y2_, z2_}, Thic_] := {
   Thickness[Thic],
   Line[{{x1, y1, z1}, {x2, y1, z1}, {x2, y2, z1}, {x1, y2, z1}}],
   Line[{{x1, y1, z1}, {x2, y1, z1}, {x2, y1, z2}, {x1, y1, z2}}],
   Line[{{x2, y1, z1}, {x2, y2, z1}, {x2, y2, z2}, {x2, y1, z2}}],
   Line[{{x2, y2, z1}, {x1, y2, z1}, {x1, y2, z2}, {x2, y2, z2}}],
   Line[{{x1, y2, z1}, {x1, y1, z1}, {x1, y1, z2}, {x1, y2, z2}}],
   Line[{{x1, y1, z2}, {x2, y1, z2}, {x2, y2, z2}, {x1, y2, z2}}]};
```

Fonction réalisant une sphère non coupée en indiquant le rayon et la couleur

```
FormeSphere[R_, Col_] := {
   EdgeForm[],
   SurfaceColor[Apply[RGBColor, Col]],
   Sphere[R, 40, 30]};
```

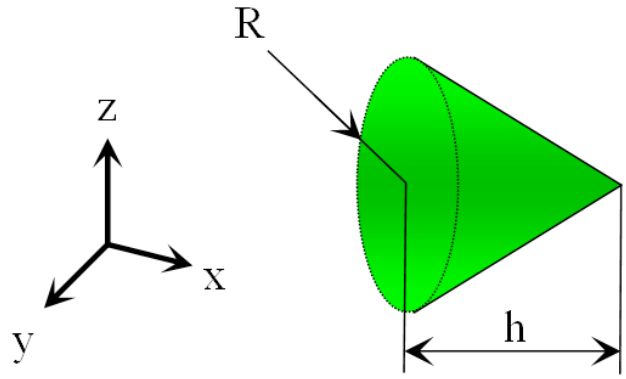


Fonction réalisant une sphère en indiquant la partie à couper, le rayon et la couleur

```
FormeSphereCoupee[Drp_, R_, Col_] := {
   EdgeForm[],
   SurfaceColor[Apply[RGBColor, Col]],
   Drop[Sphere[R, 40, 30], Drp]};
```

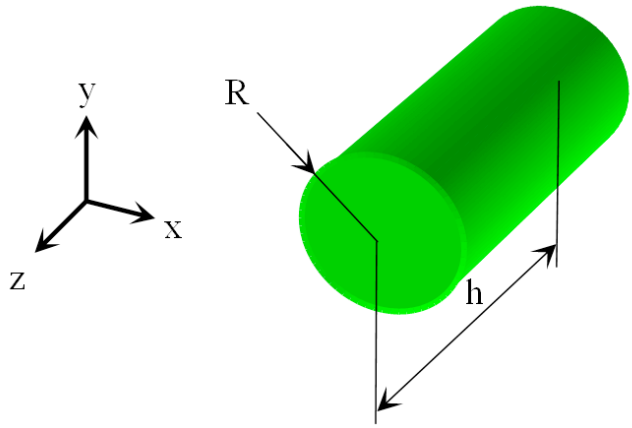
Fonction réalisant un cone en indiquant le rayon, la demi hauteur et la couleur

```
FormeCone[R_, H_, Col_] := {
   EdgeForm[],
   SurfaceColor[Apply[RGBColor, Col]],
   Cone[R, H, 20]};
```



Fonction réalisant un cylindre en indiquant le rayon, la demi hauteur et la couleur

```
FormeCylindre[R_, H_, Col_] := {
   EdgeForm[],
   SurfaceColor[Apply[RGBColor, Col]],
   Cylinder[R, H, 45]};
```

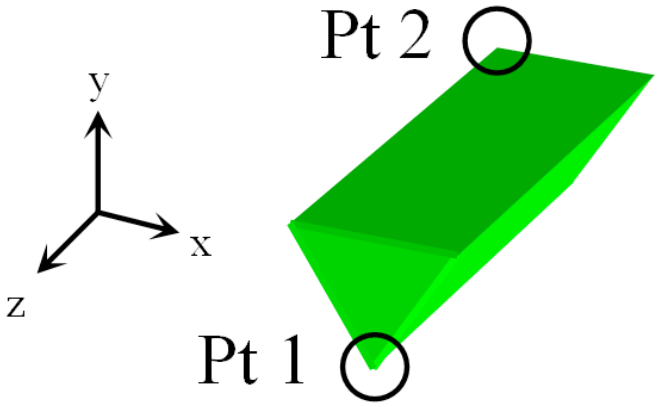


Fonction réalisant un cylindre coupé en indiquant le rayon, la demi hauteur, la couleur et la partie à couper

```
FormeCylindreCoupe[R_, H_, Col_, Drp_] := {
   EdgeForm[],
   SurfaceColor[Apply[RGBColor, Col]],
   Drop[Cylinder[R, H, 30], Drp]};
```

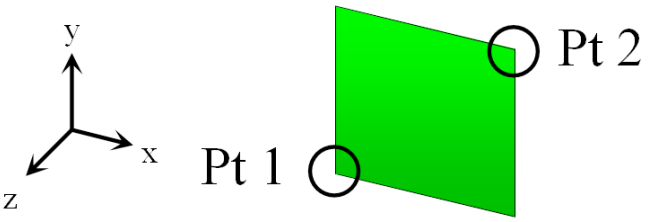
Fonction réalisant un prisme droit à base triangulaire

```
FormePrisme[{x1_, y1_, z1_}, {x2_, y2_, z2_}, Col_, Thic_] := {
   EdgeForm[{GrayLevel[0], Thickness[Thic]}],
   SurfaceColor[Apply[RGBColor, Col]],
   Polygon[{{x1, y1, z1}, {x2, y2, z1}, {2 * x1 - x2, y2, z1}}],
   Polygon[{{x1, y1, z1}, {x2, y2, z1}, {x2, y2, z2}, {x1, y1, z2}}],
   Polygon[{{x1, y1, z1}, {2 * x1 - x2, y2, z1}, {2 * x1 - x2, y2, z2}, {x1, y1, z2}}],
   Polygon[{{x2, y2, z1}, {2 * x1 - x2, y2, z1}, {2 * x1 - x2, y2, z2}, {x2, y2, z2}}],
   Polygon[{{x1, y1, z2}, {x2, y2, z2}, {2 * x1 - x2, y2, z2}}]};
```



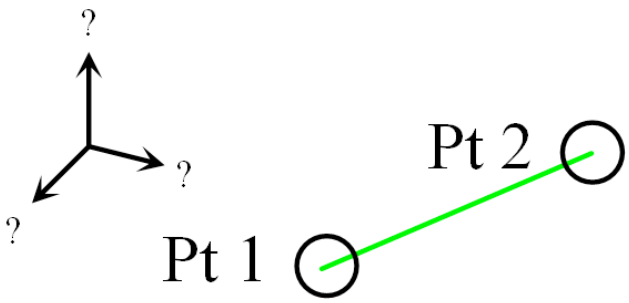
Fonction réalisant un plan porté par (x,y)

```
FormePlan[{x1_, y1_, z1_}, {x2_, y2_, z2_}, Col_, Thic_] := {
   EdgeForm[{GrayLevel[0], Thickness[Thic]}],
   SurfaceColor[Apply[RGBColor, Col]],
   Polygon[{{x1, y1, z1}, {x1, y2, z1}, {x2, y2, z1}, {x2, y1, z1}}]};
```



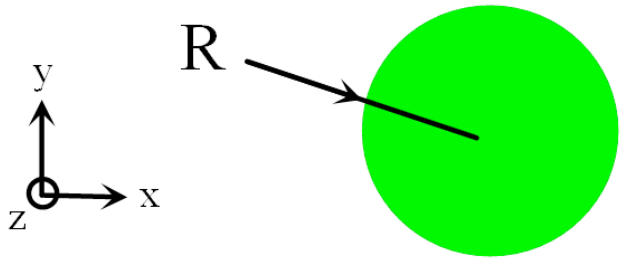
Fonction réalisant un segment d'après 2 points, de couleur et d'épaisseur désirée

```
FormeSegment[Pt1_, Pt2_, Col_, Thic_] := {
   Apply[RGBColor, Col],
   Thickness[Thic],
   Line[{Pt1, Pt2}]};
```



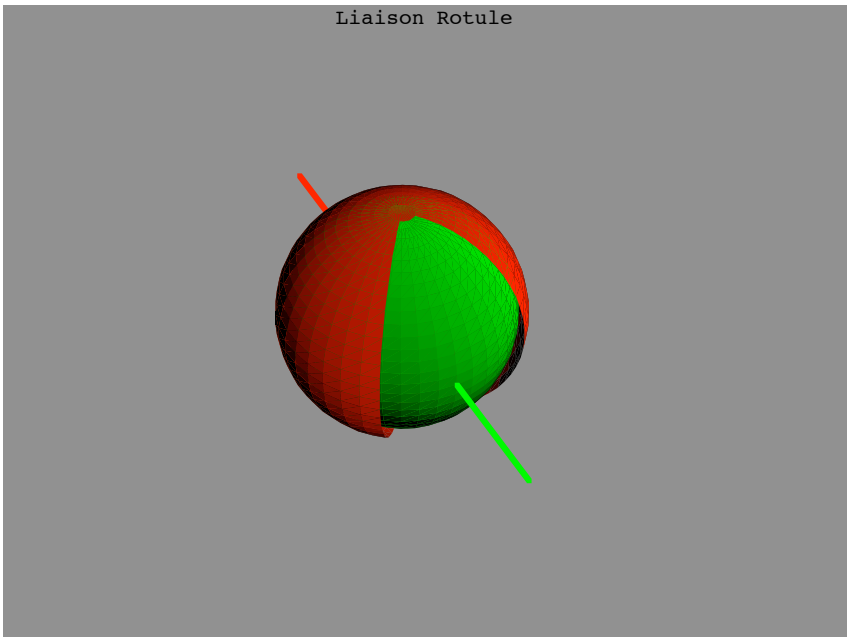
Fonction réalisant un disque dans un espace 3D, il est porté par le plan x,y

```
FormeDisque[R_, {o1_, o2_, o3_}, Col_] := {
   SurfaceColor[Apply[RGBColor, Col]],
   EdgeForm[],
   Table[
    Polygon[
     {{R * Cos [ϕ] + o1, R * Sin [ϕ] + o2, o3}, {R * Cos [ϕ + 0.1] + o1, R * Sin[ϕ + 0.1] + o2, o3},
      {o1, o2, o3}}],
    {ϕ, 0, 2 π, 0.1}]};
```



Liaison rotule

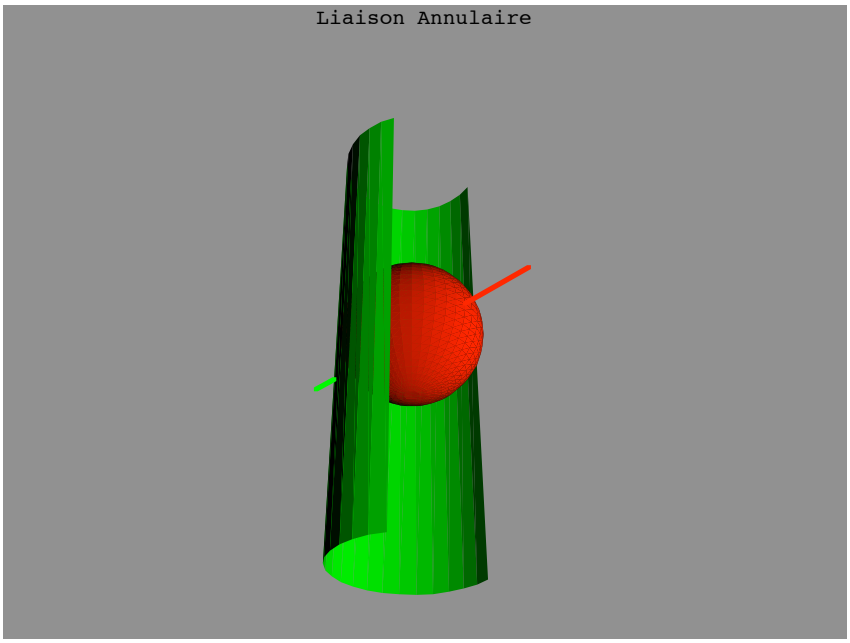
```
GraphicForm[Rotule[t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     FormeSphereCoupee[{449, 700, 1}, 2.5, Col2],
     FormeSphere[2.3, Col1],
     FormeSegment[{2.5, 0, 0}, {5, 0, 0}, Col2, 0.01],
     FormeSegment[{-2.3, 0, 0}, {-5, 0, 0}, Col1, 0.01]}, rα, rβ, rγ], t];

Show[Graphics3D[
  GraphicForm[Rotule[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
  ViewPoint → {-2, 1, 2},
  Boxed → False,
  Background -> RGBColor[0.5, 0.5, 0.5],
  ImageSize → {400, 300},
  PlotLabel -> "Liaison Rotule"]]
```



- Graphics3D -

Liaison linéaire annulaire

```
GraphicForm[Annulaire[t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     FormeSphere[2.4, Col2],
     FormeCylindreCoupe[2.5, 8, Col1, {10, 20, 1}],
     FormeSegment[{2.5, 0, 0}, {5, 0, 0}, Col1, 0.01],
     FormeSegment[{-2.4, 0, 0}, {-5, 0, 0}, Col2, 0.01]}, rα, rβ, rγ], t];

Show[Graphics3D[
  GraphicForm[Annulaire[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
  ViewPoint → {-1, 1, -1},
  Boxed → False,
  Background -> RGBColor[0.5, 0.5, 0.5],
  ImageSize → {400, 300},
  PlotLabel -> "Liaison Annulaire"]]
```

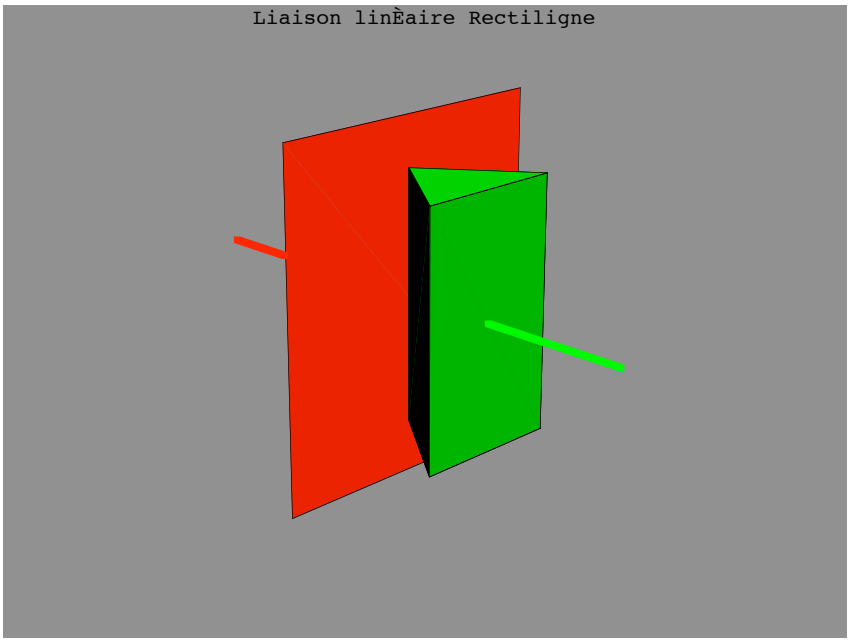


- Graphics3D -

Liaison linéaire rectiligne

```
GraphicForm[Rectiligne[t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     FormePrisme[{0, 0, 0}, {-1.5, 2, 5}, Col1, 0.001],
     RotateShape[
      FormePlan[{-3, -1, 0}, {3, 6, 0}, Col2, 0.001],
      0, -π / 2, 0],
     FormeSegment[{0, 0, 2.5}, {0, -5, 2.5}, Col2, 0.01],
     FormeSegment[{0, 2, 2.5}, {0, 5, 2.5}, Col1, 0.01]}, rα, rβ, rγ], t];
```

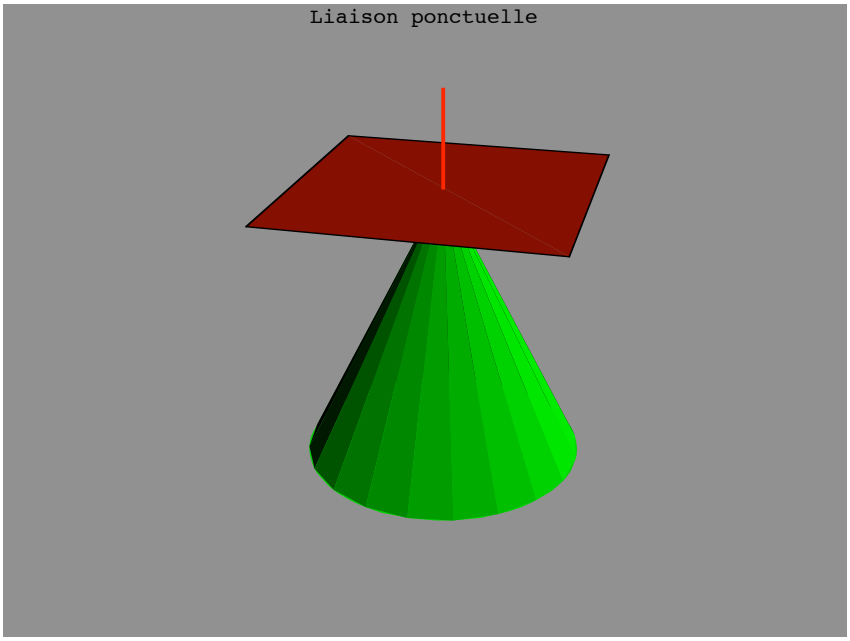
```
Show[Graphics3D[
  GraphicForm[Rectiligne[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
  ViewPoint → {2, 2, 1},
  Boxed → False,
  Background -> RGBColor[0.5, 0.5, 0.5],
  ImageSize → {400, 300},
  PlotLabel -> "Liaison linéaire Rectiligne"]]
```




```
- Graphics3D -
```

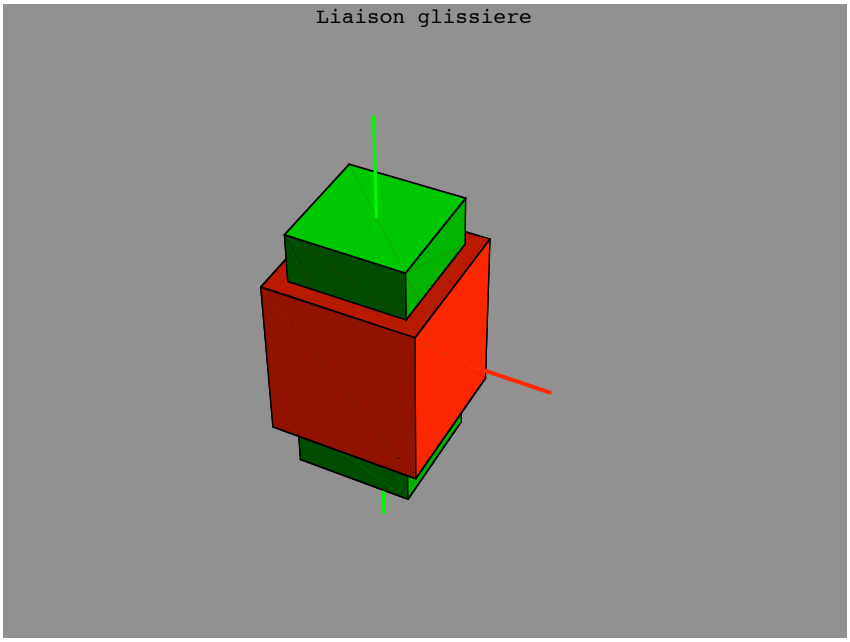
Liaison ponctuelle

```
GraphicForm[Ponctuelle[t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     RotateShape[{
       FormeCone[3, 3, Col1],
       FormeSegment[{0, 0, 2}, {0, 0, -5}, Col1, 0.01]
      }, 0, 0, 0],
     FormePlan[{3, 3, 3}, {-3, -3, 3}, Col2, 0.004],
     FormeSegment[{0, 0, 3}, {0, 0, 5}, Col2, 0.01],
     FormeDisque[3, {0, 0, -3}, Col1]
    }, rα, rβ, rγ], t];

Show[Graphics3D[
   GraphicForm[Ponctuelle[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
   ViewPoint → {2, 0.5, 1},
   Boxed → False,
   Background -> RGBColor[0.5, 0.5, 0.5],
   ImageSize → {400, 300},
   PlotLabel -> "Liaison ponctuelle"]];
```



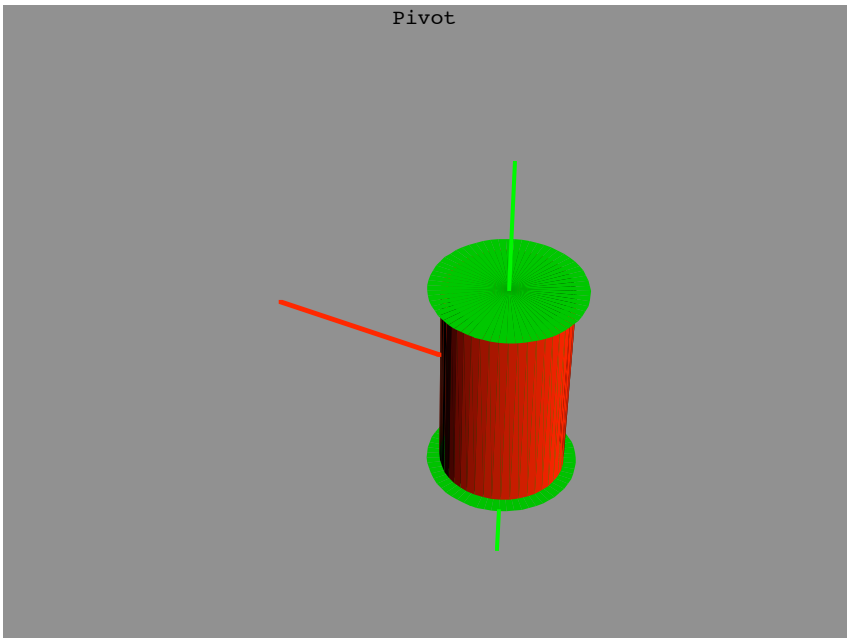
Liaison glissiere

```
GraphicForm[Glissiere[t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     FormePave[{1.7, 1.7, 4}, {-1.7, -1.7, -4}, Col1, 0.004],
     FormeSegment[{0, 0, -7}, {0, 0, 7}, Col1, 0.01],
     FormePave[{2.2, 2.2, 2.5}, {-2.2, -2.2, -2.5}, Col2, 0.004],
     FormePaveFilaire[{1.7, 1.7, 2.5}, {-1.7, -1.7, -2.5}, 0.004],
     FormeSegment[{0, 2, 0}, {0, 5, 0}, Col2, 0.01]}, rα, rβ, rγ], t];

Show[Graphics3D[
   GraphicForm[Glissiere[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
   ViewPoint → {2, 1, 2},
   Boxed → False,
   Background -> RGBColor[0.5, 0.5, 0.5],
   ImageSize → {400, 300},
   PlotLabel -> "Liaison glissiere"]];
```



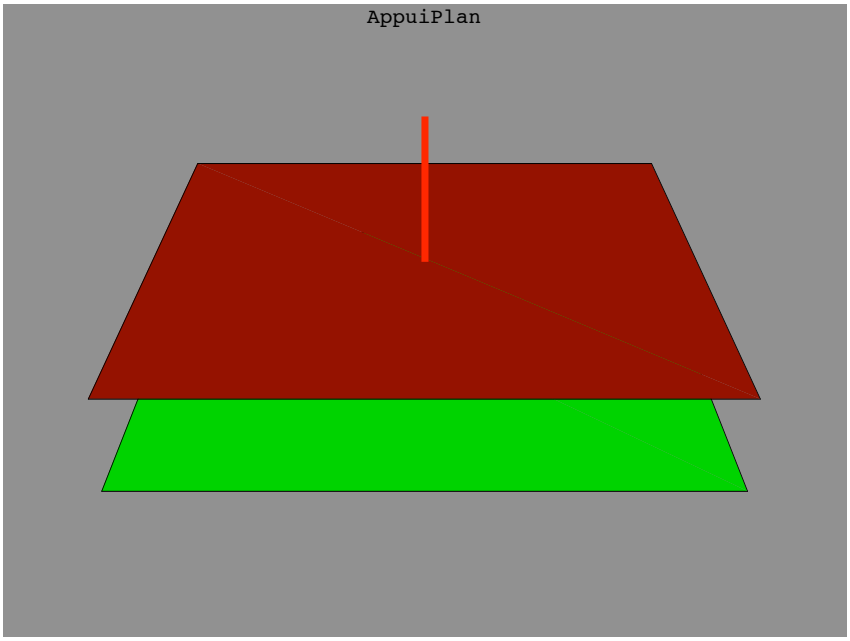
Liaison pivot

```
GraphicForm[Pivot[t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     FormeCylindre[2, 3.4, Col2],
     FormeCylindre[1.8, 3.5, Col1],
     FormeSegment[{0, -8, 0}, {0, -2, 0}, Col2, 0.01],
     FormeSegment[{0, 0, 8}, {0, 0, -8}, Col1, 0.01],
     FormeDisque[2.4, {0, 0, 3.5}, Col1],
     FormeDisque[2.4, {0, 0, -3.5}, Col1]}, rα, rβ, rγ], t];

Show[Graphics3D[
   GraphicForm[Pivot[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
   ViewPoint → {2, 1, 2},
   Boxed → False,
   Background -> RGBColor[0.5, 0.5, 0.5],
   ImageSize → {400, 300},
   PlotLabel -> "Pivot"]];
```



Liaison appui plan

```
GraphicForm[AppuiPlan [t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     FormePlan[{-3, -3, 0}, {3, 3, 0}, Col1, 0.001],
     FormePlan[{-3, -3, 1}, {3, 3, 0.5}, Col2, 0.001],
     FormeSegment[{0, 0, 1}, {0, 0, 5}, Col2, 0.01],
     FormeSegment[{0, 0, 0}, {0, 0, -5}, Col1, 0.01]}, rα, rβ, rγ], t];

Show[Graphics3D[
   GraphicForm[AppuiPlan[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
   ViewPoint → {2, 0, 1},
   Boxed → False,
   Background -> RGBColor[0.5, 0.5, 0.5],
   ImageSize → {400, 300},
   PlotLabel -> "AppuiPlan"]];
```



Liaison pivot glissant

```
GraphicForm[PivotGlissant[t_, {rα_, rβ_, rγ_}], Col1_, Col2_] :=
  TranslateShape[
   RotateShape[{
     FormeCylindre[2, 3, Col2],
     FormeCylindre[1.8, 3.5, Col1],
     FormeSegment[{0, -5, 0}, {0, -2, 0}, Col2, 0.01],
     FormeSegment[{0, 0, 5}, {0, 0, -5}, Col1, 0.01],
     FormeDisque[1.8, {0, 0, 3.5}, Col1],
     FormeDisque[1.8, {0, 0, -3.5}, Col1]}, rα, rβ, rγ], t];

Show[Graphics3D[
   GraphicForm[PivotGlissant[{0, 0, 0}, {0, 0, 0}], {0, 1, 0}, {1, 0, 0}],
   ViewPoint → {2, 1, 2},
   Boxed → False,
   Background -> RGBColor[0.5, 0.5, 0.5],
   ImageSize → {400, 300},
   PlotLabel -> "Pivot Glissant"]];
```

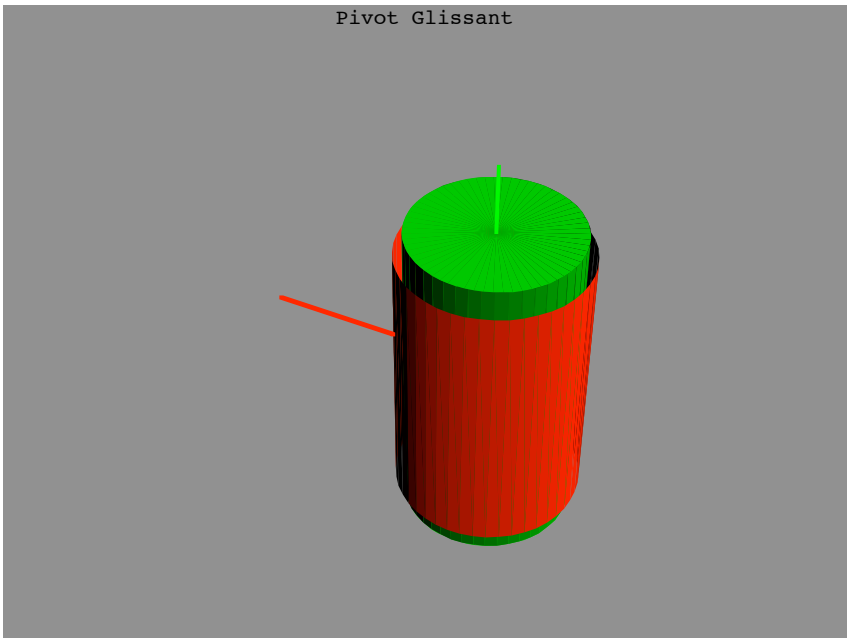
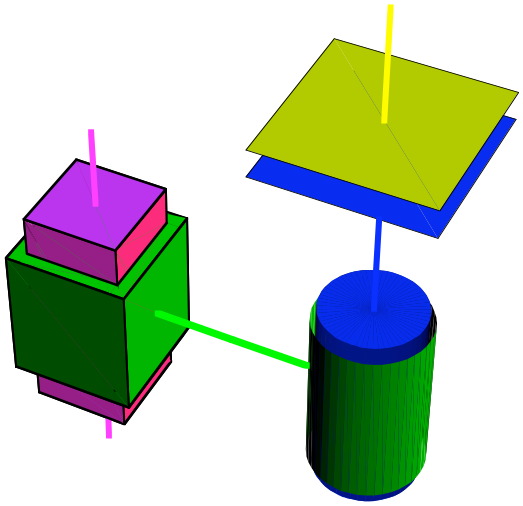


Schéma cinématique

```
Couleur[r_, s_] :=
  Part[Cases[s, Vertex[r, Body[m_, c_]] → c], 1];

Auxiliary[Joint[{s1_, s2_}, link_], s_] :=
  GraphicForm[link, Couleur[s1, s], Couleur[s2, s]];

GraphicForm[MechanicalSystem[s_]] :=
  Show[Graphics3D[Map[Auxiliary[#, s] &, Cases[s, _Joint]]],
   PlotRange → All, Boxed -> False,
   ViewPoint → {2, 1, 2},
   ImageSize → {400, 400},
   PlotRange -> All,
   PlotLabel -> "MechanicalSystem"
   (*,Axes→True,AxesLabel→{x,y,z}*)];
```

```
ms = GraphicForm[MechanicalSystem[{
     Vertex[s0, Body[m0, {1, 0, 1}]],
     Vertex[s1, Body[m1, {0, 1, 0}]],
     Joint[{s0, s1}, Glissiere[{0, -10, 0}, {0, 0, 0}] ],
     Vertex[s2, Body[m2, {0, 0, 1}]],
     Joint[{s2, s1}, PivotGlissant[{0, 0, 0}, {0, 0, 0}] ],
     Vertex[s3, Body[m3, {1, 1, 0}]],
     Joint[{s2, s3}, AppuiPlan[{0, 0, 10}, {0, 0, 0}] ]
    }]]
```

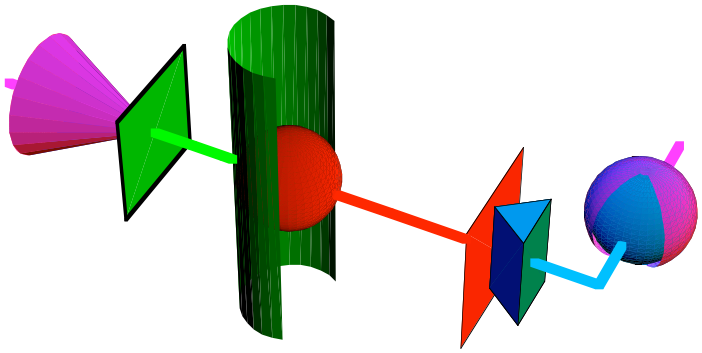
MechanicalSystem



- Graphics3D -

```
ms = GraphicForm[MechanicalSystem[{
     Vertex[s0, Body[m0, {1, 0, 1}]],
     Vertex[s1, Body[m1, {0, 1, 0}]],
     Joint[{s0, s1}, Ponctuelle[{0, 0, 0}, {0, π / 2, 0}]],
     Vertex[s2, Body[m2, {1, 0, 0}]],
     Joint[{s1, s2}, Annulaire[{0, 10, 0}, {3 π / 2, π, 0}]],
     Vertex[s3, Body[m3, {0, 0.7, 1}]],
     Joint[{s3, s2}, Rectiligne[{0, 20, -2.5}, {0, 0, 0}]],
     Vertex[s4, Body[m4, {1, 0, 1}]],
     Joint[{s3, s4}, Rotule[{-5, 25, 0}, {π, 0, 0}]]
    }]]
```

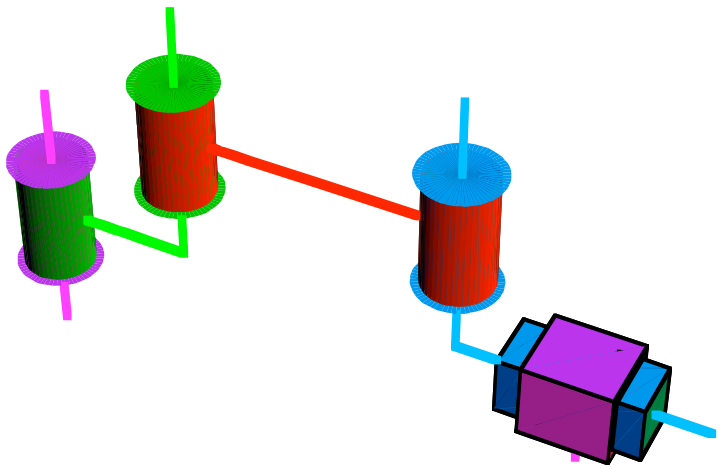
MechanicalSystem



- Graphics3D -

```
ms = GraphicForm[MechanicalSystem[{
     Vertex[s0, Body[m0, {1, 0, 1}]],
     Vertex[s1, Body[m1, {0, 1, 0}]],
     Joint[{s0, s1}, Pivot[{0, 0, 0}, {π, 0, 0}]],
     Vertex[s2, Body[m2, {1, 0, 0}]],
     Joint[{s1, s2}, Pivot[{0, 8, 8}, {π, 0, 0}]],
     Vertex[s3, Body[m3, {0, 0.7, 1}]],
     Joint[{s3, s2}, Pivot[{0, 24, 8}, {0, 0, 0}]],
     Vertex[s4, Body[m4, {1, 0, 1}]],
     Joint[{s3, s4}, Glissiere[{0, 31, 0}, {0, π / 2, 0}]]
    }]]
```

MechanicalSystem



- Graphics3D -