Documentation sur la fusion d'objets jps

JPV & ML

30 septembre 2 007
version 0.3

Résumé

Voici l'étape la plus intéressante car elle permet de composer une vue d'un ensemble d'objets de façon automatique, le calcul des parties cachées est effectué par les routines du code `PostScript` du fichier `solides.pro`. Ces routines proviennent directement du code `postscript` mis-au point par Jean-Paul Vignault sur le site de :

`http://melusine.eu.org/syracuse/bbgraf/`

Il faut donc, avant de se lancer dans cette étape, assimiler la syntaxe de ces routines.

Pour une scène composée de plusieurs objets complexes la durée d'affichage est relativement longue et c'est assurément un point qu'il convient d'améliorer. C'est pourquoi nous conseillons de créer un fichier `pdf` à partir du fichier `ps` pour une utilisation ultérieure.

1 Le code jps

1.1 Le choix de l'objet

- 4␣3␣2␣newcylindricalring : choix de l'anneau cylindrique défini par sa section dans une commande `\psset{section=\SectionPneu}` avec comme caractéristiques : `r1=4,r0=3,h=2`.
- 2␣1.5␣6␣[4␣16]␣newcylindre : choix du cylindre vertical avec comme caractéristiques :
  - `rayon=1.5` ;
  - `z0=2` est la position du centre de la base inférieure sur l'axe Oz ;
  - `z1=6` est la position du centre de la base supérieure sur l'axe Oz ;
  - `[4 16]` : le cylindre est découpé verticalement en 4 morceaux et horizontalement en 16 secteurs.

1.2 Les transformations

- `{-1␣2␣5␣translatepoint3d} solidtransform` : l'objet préalablement sélectionné subit une translation au point de coordonnées ($x = -1, y = 2, z = 5$).
- `{90␣0␣45␣rotateOpoint3d} solidtransform` : l'objet préalablement sélectionné subit une rotation autour des axes (Ox,Oy,Oz), dans cet ordre, de 90° autour de (Ox) suivie d'une rotation de 45° autour de (Oz).

1.3 Le choix de la couleur de l'objet

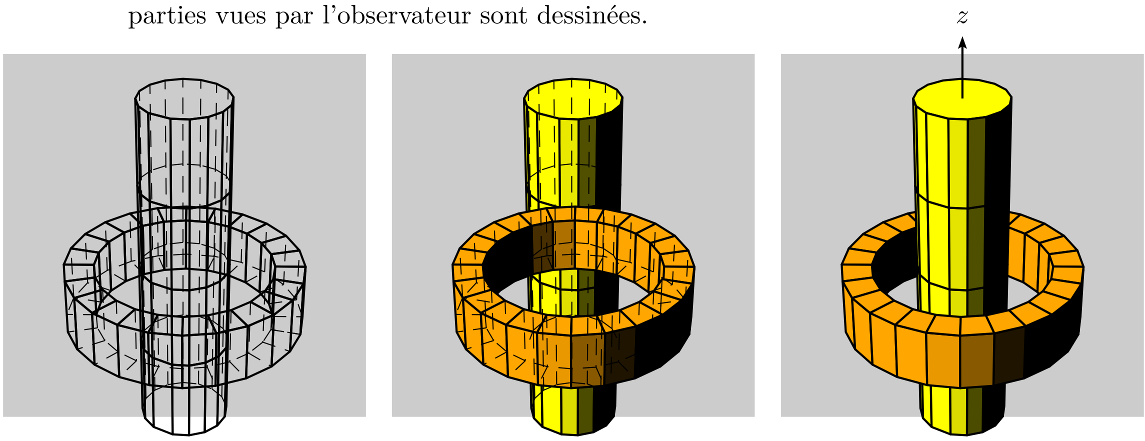
- dup (jaune) outputcolors : l'objet de couleur jaune éclairé en lumière blanche.

1.4 La fusion des objets

– Elle se fait avec l'instruction `solidfuz`.

1.5 Le dessin des objets

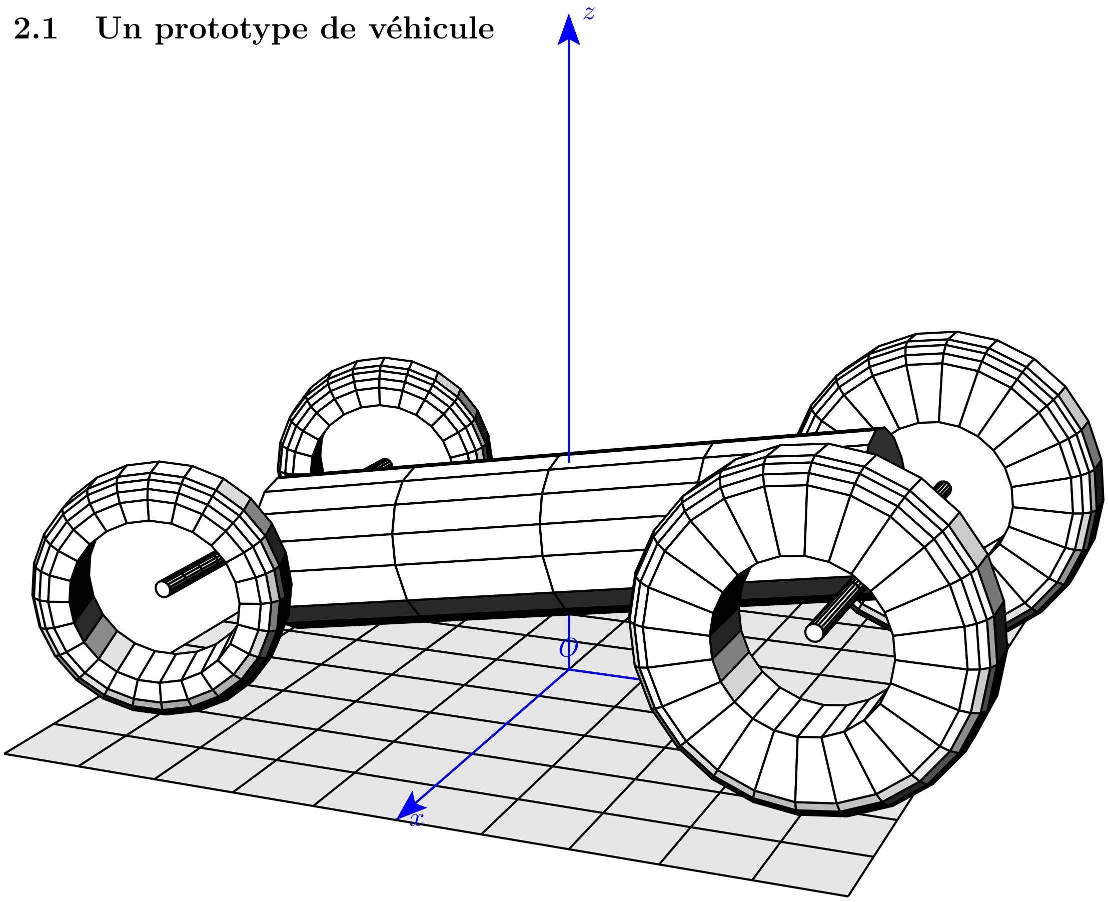
– Le tracé a trois options :
  – `drawsolid` : tracé des arêtes uniquement, les arêtes cachées sont en pointillés ;
  – `drawsolid*` : tracé et remplissage des solides dans l'ordre de leur programmation (option peu intéressante à priori), avec le dessin des arêtes cachées en pointillés ;
  – `drawsolid**` : tracé et remplissage des solides avec l'algorithme du peintre : seules les parties vues par l'observateur sont dessinées.




```
\psset{lightsrc=50 -50 50,SphericalCoor=true,viewpoint=50 20 50,Decran=50}
\begin{pspicture}(-6,-2)(6,8)
\psframe(-6,-2)(6,8)
\codejps{
% solide 1
 /tour {
    -6 1.5 6 [4 16] newcylindre
    dup (jaune) outputcolors
    } def
% solide 2
 /anneau {
    4 3 2 newcylindricalring
   {0 0 -1 translatepoint3d} solidtransform
    dup (orange) outputcolors
    } def
% fusion
    tour anneau solidfuz
    drawsolid**}
\end{pspicture}
```

2 Scènes complexes

2.1 Un prototype de véhicule



Il faut opérer en plusieurs étapes en fusionnant les solides deux par deux.

– On fusionne d'abord les deux roues avant `roue12`:

```
  /roue12 {
% solide 1
    2 1 1 newcylindricalring
     {90 0 90 rotateOpoint3d} solidtransform
     {3 4 2 translatepoint3d} solidtransform
     dup (White) outputcolors
% solide 2
    2 1 1 newcylindricalring
    {90 0 90 rotateOpoint3d} solidtransform
   {-3 4 2 translatepoint3d} solidtransform
    dup (White) outputcolors
% fusion
    solidfuz } def
```

– Puis ces deux roues et leur axe :

```
/axe12{
0 0.1 6  [4 16] newcylindre
{90 0 90 rotateOpoint3d} solidtransform
{-3 4 2  translatepoint3d} solidtransform
dup (White) outputcolors
} def
/roue12axes {
roue12 axe12 solidfuz } def
```

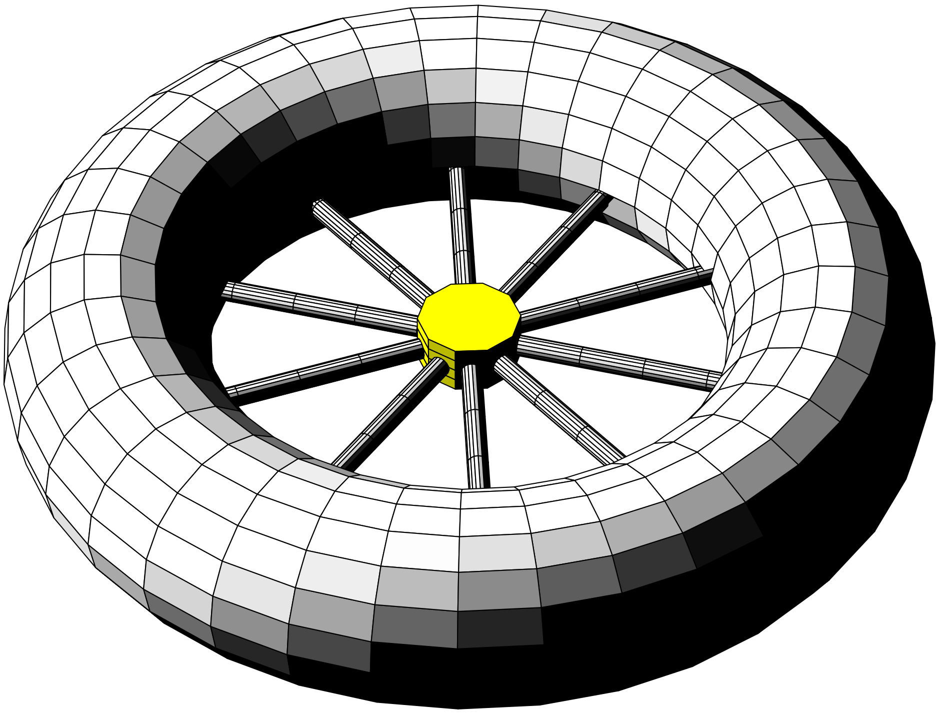
– On opère de même pour les roues arrière et leur axe :

```
/roue34 {
% solide 3
    1.5 1 1 newcylindricalring
    {90 0 110 rotateOpoint3d} solidtransform
   {3 -4 1.5 translatepoint3d} solidtransform
    dup (White) outputcolors
% solide 4
    1.5 1 1 newcylindricalring
   {90 0 110 rotateOpoint3d} solidtransform
   {-3 -4 1.5 translatepoint3d} solidtransform
    dup (White) outputcolors
% fusion
    solidfuz } def
/axe34{
0 0.1 6 [16 16] newcylindre
{90 0 90 rotateOpoint3d} solidtransform
{-3 -4 1.5  translatepoint3d} solidtransform
dup (White) outputcolors
} def
/roue34axes34 {
roue34 axe34 solidfuz } def
/roues {roue34axes34 roue12axes solidfuz} def
```

– La dernière étape consiste à fusionner les deux solides ainsi obtenus avec le semblant de chassis :

```
/chassis {
0 1 8  [4 16] newcylindre
{100 0 0 rotateOpoint3d} solidtransform
{0 4 2.5  translatepoint3d} solidtransform
dup (White) outputcolors
} def
roues chassis solidfuz
    drawsolid**}
```

2.2 Une roue ou bien une station spatiale !



On définit d'abord le premier rayon :

```
 /rayon0 {
     1 0.2 6 [4 16] newcylindre
     {90 0 0 rotateOpoint3d} solidtransform
      dup (White) outputcolors
   } def
```

Puis dans une boucle on fusionne tous les rayons de la roue :

```
36 36 360 {
    /angle exch def
  /rayon1 {
     1 0.2 6  [4 16] newcylindre
     {90 0 angle rotateOpoint3d} solidtransform
      dup (White) outputcolors
   } def
  /rayons {rayon0 rayon1 solidfuz} def
  /rayon0 rayons def
  } for
```

Ensuite, on dessine le moyeu et la circonférence (pneu) de la roue pour enfin fusionner l'ensemble :

```
 /moyeu { -0.5 1 0.5  [4 10] newcylindre dup (White) outputcolors} def
 /rayonsmoyeu {rayons  moyeu solidfuz} def
```

```
/pneu {2 7 newtore dup (jaune) outputcolors} def
/ROUE {pneu rayonsmoyeu solidfuz} def
 ROUE  drawsolid**
```