

pst-solides3d : section d'un solide par un plan avec PStricks

16 juin 2008

Résumé

Un solide peut être partagé par un plan et les deux parties obtenues être utilisées séparément. Les illustrations de ce document ont été essentiellement obtenues en employant des commandes **PStricks**, un deuxième document traite du même sujet, mais par usage du code **jps**. De nombreux autres exemples sont disponibles dans les liens donnés à la fin de document.

Table des matières

1	Tracer l'intersection d'un plan et d'un solide	2
1.1	Les paramètres	2
2	Coupes d'un solide	2
2.1	Coupe du solide plein	2
2.2	Coupe du solide creux	4
3	Tranche d'une pyramide	5
3.1	Marquage des lignes de niveau et première découpe	5
3.2	Deuxième découpe et son insertion dans la pyramide	7
4	Coupe d'un octaèdre par un plan parallèle à l'une des faces	8
4.1	Voir l'intérieur	8
4.2	On considère le solide comme plein	9
4.3	Les deux parties du solide découpé	10
5	Coupes d'un cube	11
5.1	Marquage de la ligne de découpage	11
5.2	Représentation du cube découpé avec une face de découpe hexagonale	12
5.3	Cube découpé dans des différentes positions	13
6	Sections multiples	14
6.1	Coupes dans une sphère avec PStricks	14
6.2	Sections multiples d'un parallélépipède	15
7	Sections d'un tore	17
8	Autres exemples	17

1 Tracer l'intersection d'un plan et d'un solide

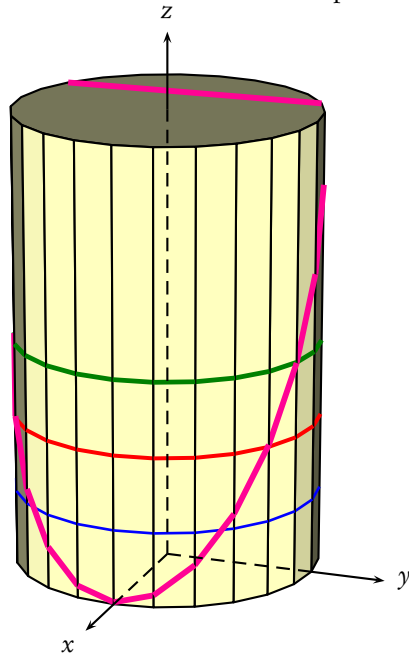
1.1 Les paramètres

C'est l'option `intersectionplan={ [a b c d] }` qui permet de tracer l'intersection d'un plan et d'un solide. L'argument entre les crochets contient les paramètres du plan affine ayant pour équation : $ax + by + cz + d = 0$. Il est possible de dessiner l'intersection du solide avec plusieurs plans en plaçant à la suite les paramètres de ces plans comme dans l'exemple suivant.

Le tracé est activé avec `intersectiontype=0` ou toute autre valeur ≥ 0 .

La couleur du tracé est choisie dans l'option `intersectioncolor=(bleu) (rouge) etc.` et l'épaisseur par `intersectionlinewidth=1 2 etc.` On donne successivement dans l'ordre, l'épaisseur de chaque tracé en points.

Le tracé en traits discontinus des parties cachées sera activé avec `action=draw`.



```
\begin{pspicture}(-3,-2)(3,7)
\psset{SphericalCoor=true,viewpoint=50 20 20,Decran
=50}
\lightsource
\psSolid[object=cylindre,
ngrid=1 24,
r=2,
fillcolor=yellow!25,
intersectiontype=0,
intersectionplan={
[0 0 1 -1]
[0 0 1 -2]
[0 0 1 -3]
[0.894 0 0.447 -1.8]},
intersectioncolor=(bleu) (rouge) (vert) (rose),
intersectionlinewidth=1 1.5 1.8 2.2]
\axesIIIID(2,2,6)(3,3,7)
\end{pspicture}
```

2 Coupes d'un solide

2.1 Coupe du solide plein

L'objet étudié est un cylindre. Le plan qui coupe l'objet sera défini par :

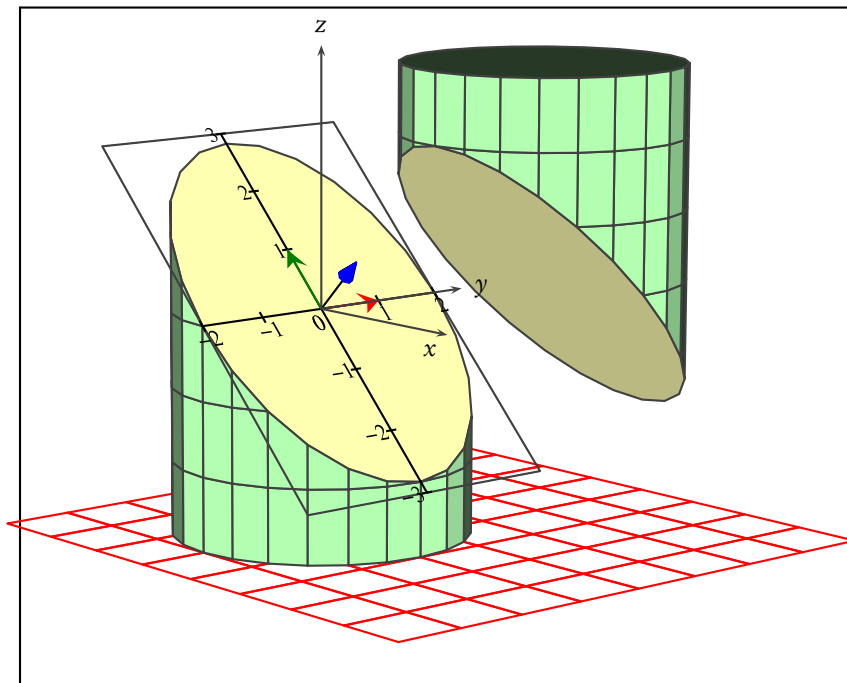
```
plansepare={ [a b c d] }
```

Les deux parties ne seront pas tracées mais mises en mémoire avec le nom commun `name=partiescylindre` :

```
\psset{solidmemory}  
\psSolid[object=cylindre,  
  r=2,h=6  
  ngrid=6 24,  
  plansepare={[0.707 0 0.707 0]},  
  name=partiescylindre,  
  action=none](0,0,-3)
```

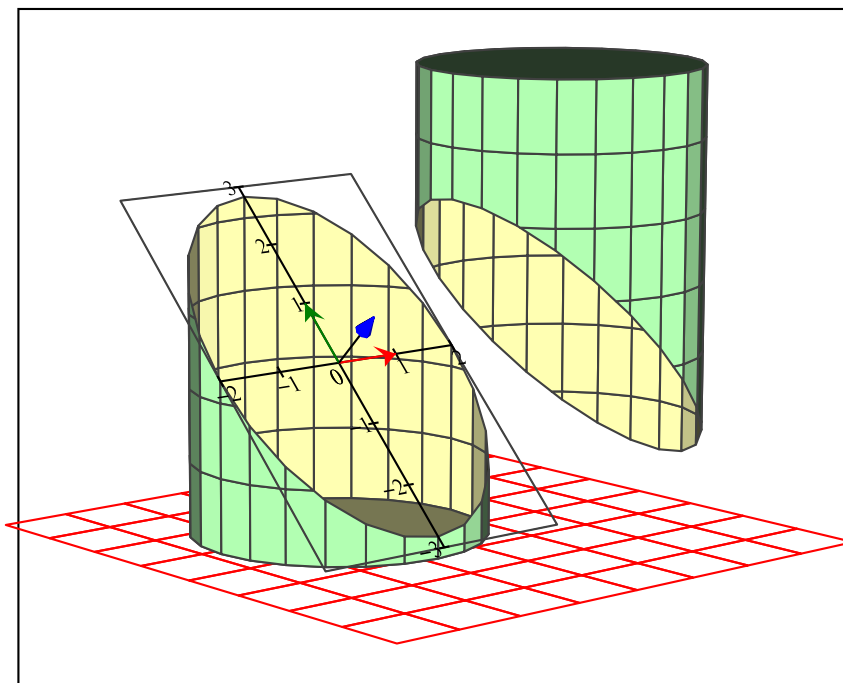
Puis affichées séparément avec leur indice respectif. C'est le sens de la normale du plan de séparation qui détermine le numérotage des deux parties : `0` celle qui est *au-dessus* de la normale et `1` celle qui est *au-dessous*. Pour les deux parties, la face de découpe porte le numéro `0`. S'il y a plusieurs faces de découpe, comme dans le cas du tore elles sont numérotées `0, 1 etc.`

```
\psSolid[object=load,  
  load=partiescylindre1,  
  fillcolor={[rgb]{0.7 1 0.7 }},  
  fcol=0 (1 1 0.7 setrgbcolor)]  
\psSolid[object=load,  
  load=partiescylindre0, RotZ=60,  
  fillcolor={[rgb]{0.7 1 0.7 }},  
  fcol=0 (1 1 0.7 setrgbcolor)](0,4,0)
```



2.2 Coupe du solide creux

Les options `rm=0, hollow` permettent, l'une d'enlever la face de découpe `rm=0` et l'autre, `hollow` de voir l'intérieur.



3 Tranche d'une pyramide

3.1 Marquage des lignes de niveau et première découpe

Cette pyramide est créée comme un `object=new` en donnant la liste des coordonnées des sommets et des faces.

```
sommets=
  0 -2 0 %% 0
  -2 0 0 %% 1
  0 4 0 %% 2
  4 0 0 %% 3
  0 0 5, %% 4
faces={
  [3 2 1 0]
  [4 0 3]
  [4 3 2]
  [4 2 1]
}
```

Dans une première étape, facultative, on marque les lignes de découpe.

```
intersectiontype=0,
intersectionplan={[0 0 1 -1]
                 [0 0 1 -2]},
intersectionlinewidth=1 2,
intersectioncolor=(bleu) (rouge)
```

Et on coupe la pointe supérieure, en dessinant aussi le plan de coupe.

```
\psSolid[object=new,
sommets=
  0 -2 0 %% 0
  -2 0 0 %% 1
  0 4 0 %% 2
  4 0 0 %% 3
  0 0 5, %% 4
faces={
  [3 2 1 0]
  [4 0 3]
  [4 3 2]
  [4 2 1]
  [4 1 0]},
```

```

    plansepare={[0 0 1 -2]},
    name=firstSlice,
    action=none]
\psSolid[object=load,action=draw*,
    load=firstSlice1]
\psSolid[object=plan,
    definition=equation,
    args={[0 0 1 -2]},
    base=-3 5 -3 5,action=draw]

```

Pour ne pas avoir à chaque fois à réécrire sommets et faces de la pyramide, on enregistre ses données dans les fichiers :

- Pyramid-couleurs.dat
- Pyramid-faces.dat
- Pyramid-sommets.dat
- Pyramid-io.dat

grâce à la commande `action=writesolid` :

```

\psSolid[object=new,
    sommets=
        0 -2 0 %% 0
        -2 0 0 %% 1
        0 4 0 %% 2
        4 0 0 %% 3
        0 0 5, %% 4
    faces={
        [3 2 1 0]
        [4 0 3]
        [4 3 2]
        [4 2 1]
        [4 1 0]
    },file=Pyramid,fillcolor=yellow!50,
    action=writesolid]

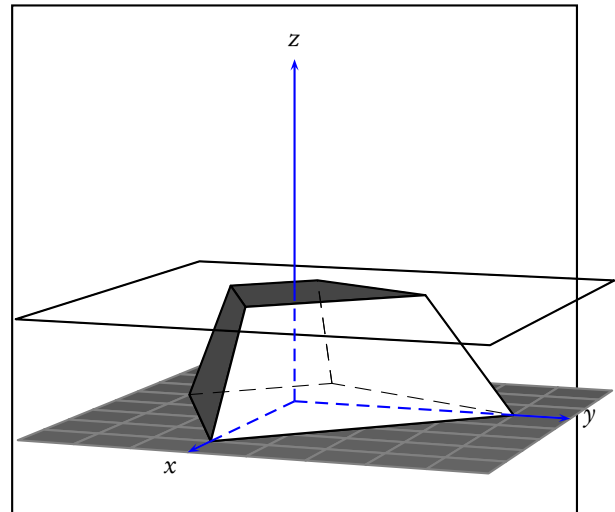
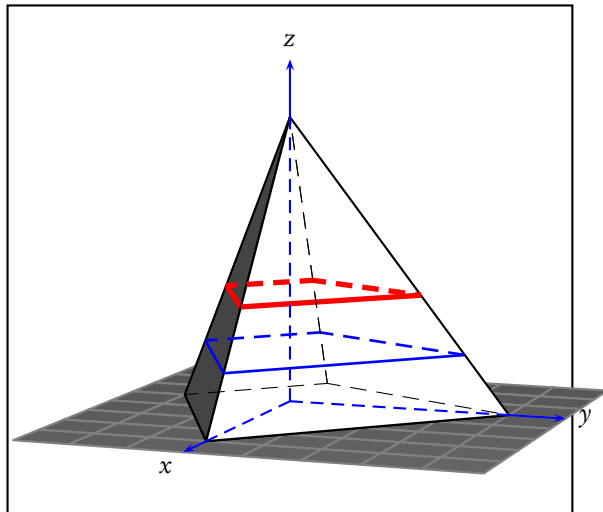
```

Toutes ces lignes pourront alors être supprimées et par la suite, on appellera ces données avec la commande :

```

\psSolid[object=datfile,
    file=Pyramid]

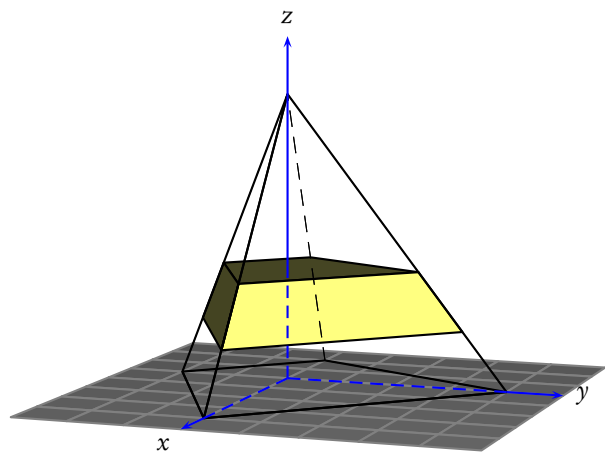
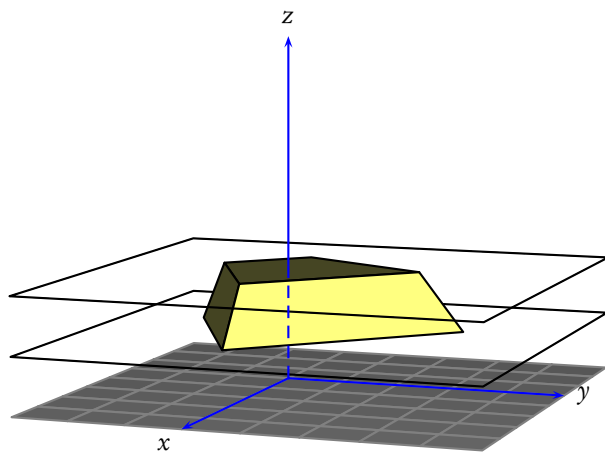
```



3.2 Deuxième découpe et son insertion dans la pyramide

Après avoir ôté la partie supérieure `firstSlice0` (elle n'apparaît plus), on découpe dans la partie restante `firstSlice1`, la base de la pyramide en gardant le haut `secondSlice0`, puis on enregistre la tranche de pyramide restante afin de l'insérer dans la pyramide en fil de fer :

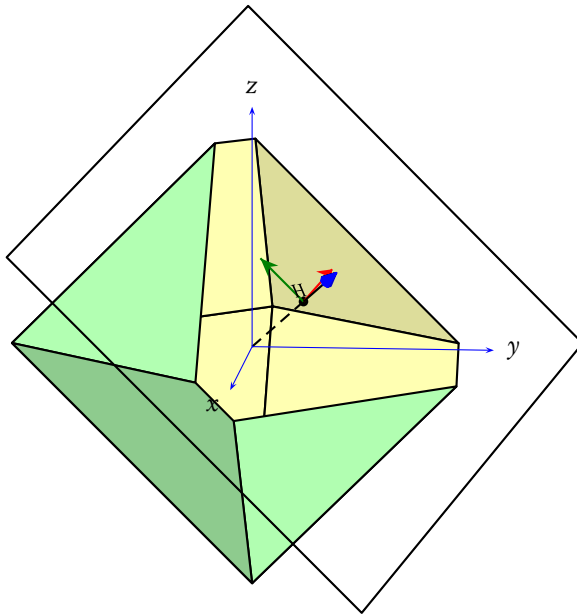
```
\psset{solidmemory}
\psSolid[object=datfile,
    file=Pyramid,
    plansepare={[0 0 1 -2]},
    name=firstSlice,
    action=none]
\psSolid[object=load,
    load=firstSlice1,
    action=none,
    plansepare={[0 0 1 -1]},
    name=secondSlice]
\psSolid[object=load,action=draw*,
    load=secondSlice0]
\psSolid[object=load,
    load=secondSlice0,
    file=slicePyramid,
    action=writesolid]
\psSolid[object=datfile,fillcolor=yellow!50,
    file=slicePyramid]
```



4 Coupe d'un octaèdre par un plan parallèle à l'une des faces

4.1 Voir l'intérieur

Rappelons que ce sont les options `rm=0,hollow` qui permettent, l'une d'enlever la face de découpe `rm=0` et l'autre `hollow` de voir l'intérieur.



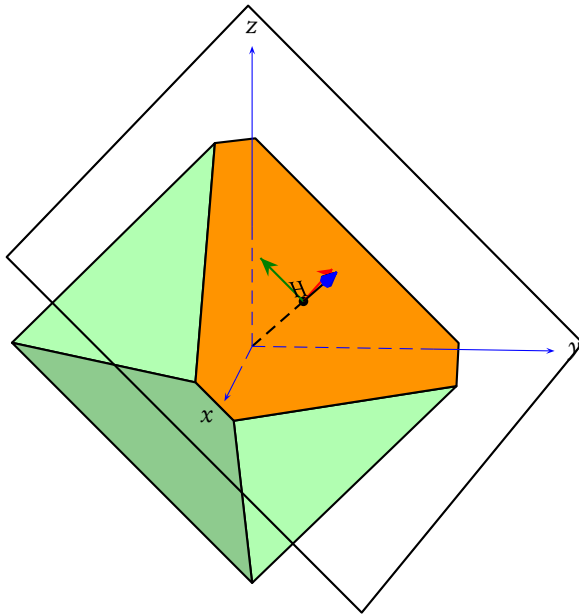
```

1 \begin{pspicture}(-3.5,-3)(4.5,5)
2 \psset{SphericalCoor=true,viewpoint=100 5
3   10,Decran=80,a=4}
4 \lightsource
5 \pstVerb{/distance 0.4 4 mul def % 0.4*a
6   /n_x 45 cos 35.2644 cos mul def
7   /n_y 45 sin 35.2644 cos mul def
8   /n_z 35.2644 sin def
9   /xH distance n_x mul def
10  /yH distance n_y mul def
11  /zH distance n_z mul def
12  }%
13 \psset{solidmemory}
14 \psSolid[object=octahedron,
15   plansepare={
16     [ n_x % a
17       n_y % b
18       n_z % c
19       distance neg % d
20     ]},
21   name=part,
22   action=none]
23 \psSolid[object=load,
24   rm=0,
25   hollow,
26   fillcolor={[rgb]{0.7 1 0.7}},
27   incolor={[rgb]{1 1 0.7}},
28   load=part1]
29 \psPoint(xH,yH,zH){H}
30 \psPoint(0,0,0){O}\psdot(H)
31 \psline[linestyle=dashed](O)(H)
32 \psSolid[object=plan,
33   definition=normalpoint,
34   args={xH yH zH [n_x n_y n_z 180]},
35   base=-4 4 -4 4,action=draw,showBase]
36 \psProjection[object=texte,
37   fontsize=10,pos=dc,text=H,
38   normal=n_x n_y n_z -30](xH,yH,zH)
39 \axesIIID[linecolor=blue,linewidth=0.4pt
40   ](0,0,0)(4,4,4)
41 \end{pspicture}

```

4.2 On considère le solide comme plein

L'option `fcol=0 (YellowOrange)` permet de colorier la face de découpe qui est la face `0`.



```

1 \begin{pspicture}(-3.5,-3)(4.5,5)
2 \psset{SphericalCoor=true,viewpoint=100 5
3   10,Decran=80,a=4}
4 \lightsource
5 \pstVerb{/distance 0.4 4 mul def % 0.4*a
6   /n_x 45 cos 35.2644 cos mul def
7   /n_y 45 sin 35.2644 cos mul def
8   /n_z 35.2644 sin def
9   /xH distance n_x mul def
10  /yH distance n_y mul def
11  /zH distance n_z mul def
12  }%
13 \psset{solidmemory}
14 \psSolid[object=octahedron,
15   plansepare={
16     [ n_x % a
17       n_y % b
18       n_z % c
19       distance neg % d
20     ]},
21   name=part,
22   action=none]
23 \psSolid[object=load,
24   fcol=0 (YellowOrange),
25   fillcolor={[rgb]{0.7 1 0.7}},
26   load=part1]
27 \psPoint(xH,yH,zH){H}
28 \psPoint(0,0,0){O}\psdot(H)
29 \psline[linestyle=dashed](O)(H)
30 \psSolid[object=plan,
31   definition=normalpoint,
32   args={xH yH zH [n_x n_y n_z 180]},
33   base=-4 4 -4 4,action=draw,showBase]
34 \psProjection[object=texte,
35   fontsize=12,pos=dc,text=H,
36   normal=n_x n_y n_z -30](xH,yH,zH)
37 \axesIIID[linewidth=0.4pt
38   ](3,3,2)(5,5,5)
39 \end{pspicture}

```

4.3 Les deux parties du solide découpé

On rappelle que c'est le sens de la normale du plan de séparation qui détermine le numérotage des deux parties : **0** celle qui est *au-dessus* de la normale et **1** celle qui est *au-dessous*. Pour les deux parties, la face de découpe porte le numéro **0**. S'il y a plusieurs faces de découpe, comme dans le cas du tore elles sont numérotées **0, 1 etc.**

On opère en deux étapes, mise en mémoire des deux parties du solide découpé :

```

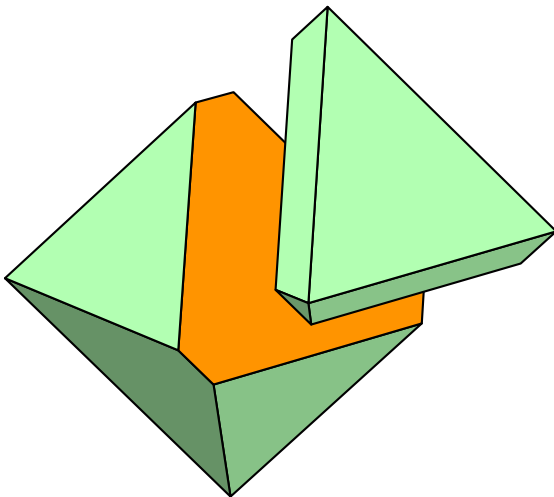
\psset{solidmemory}
\psSolid[object=octahedron,
  plansepare={
    [ n_x % a
      n_y % b
      n_z % c
      distance neg % d
    ]},

```

```
name=part,
action=none]
```

Puis placement et traitement de chacune des parties :

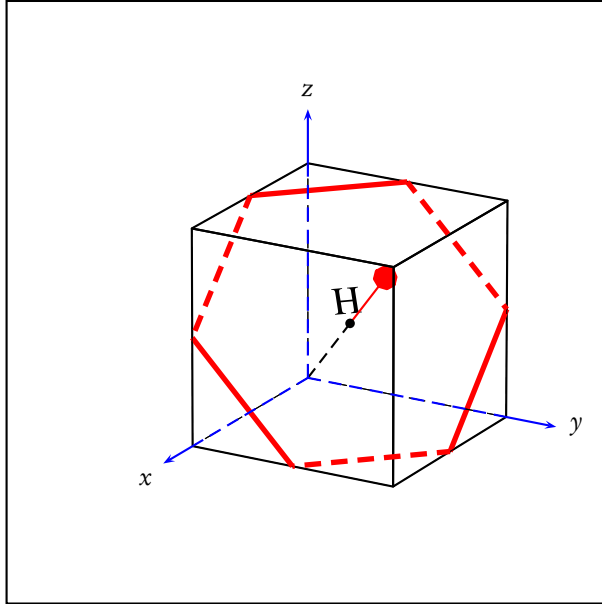
```
\psSolid[object=load,
    fcol=0 (YellowOrange),
    fillcolor={[rgb]{0.7 1 0.7}},
    load=part1]
\psSolid[object=load,
    fillcolor={[rgb]{0.7 1 0.7}},
    load=part0](2 xH mul,2 yH mul,2 zH mul)
\composeSolid
```



```
1 \begin{pspicture}(-3.5,-3)(4.5,5)
2 \psset{SphericalCoord=true,viewpoint=100 5
3   20,Decran=150,a=2}
4 \lightsource
5 \pstVerb{/distance 0.4 2 mul def % 0.4*a
6   /n_x 45 cos 35.2644 cos mul def
7   /n_y 45 sin 35.2644 cos mul def
8   /n_z 35.2644 sin def
9   /xH distance n_x mul def
10  /yH distance n_y mul def
11  /zH distance n_z mul def
12  }%
13 \psset{solidmemory}
14 \psSolid[object=octahedron,
15   plansepare={
16     [ n_x % a
17       n_y % b
18       n_z % c
19       distance neg % d
20     ]},
21   name=part,
22   action=none]
23 \psSolid[object=load,
24   fcol=0 (YellowOrange),
25   fillcolor={[rgb]{0.7 1 0.7}},
26   load=part1]
27 \psSolid[object=load,
28   fillcolor={[rgb]{0.7 1 0.7}},
29   load=part0](2 xH mul,2 yH mul,2 zH
30   mul)
31 \composeSolid
32 \end{pspicture}
```

5 Coupes d'un cube

5.1 Marquage de la ligne de découpage

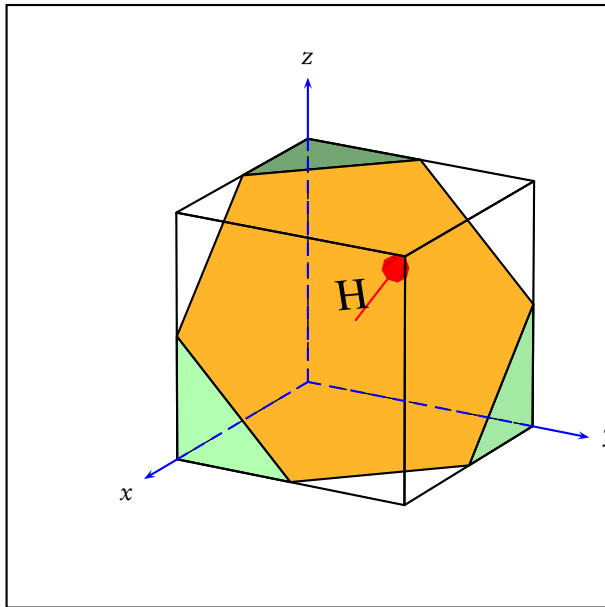


```

1 \psset{SphericalCoor=true,viewpoint=100
2   30 20,Decran=150}
3 \begin{pspicture}(-4,-3)(4,5)
4 \pstVerb{/distance 1.732 def
5   /n_x 0.57735 def
6   /n_y 0.57735 def
7   /n_z 0.57735 def
8   /xH distance n_x mul def
9   /yH distance n_y mul def
10  /zH distance n_z mul def
11  }%
12 \psframe(-4,-3)(4,5)
13 \psProjection[object=texte,
14   fontsize=10,pos=dc,text=H,
15   normal=n_x n_y n_z -30](xH,yH,zH)
16 \Normale[linecolor=red](distance
17   ,45,35.2644)
18 \psSolid[object=cube,a=2,action=draw,
19   intersectiontype=0,
20   intersectionplan={
21     [ n_x % a
22       n_y % b
23       n_z % c
24     distance neg]],
25   intersectionlinewidth=2,
26   intersectioncolor=(rouge)](1,1,1)
27 \psPoint(xH,yH,zH){H}
28 \psPoint(0,0,0){O}
29 \psline[linestyle=dashed](O)(H)
30 \psdot(H)
31 \axesIIIID[linecolor=blue](2,2,2)
32   (2.5,2.5,2.5)
33 \end{pspicture}

```

5.2 Représentation du cube découpé avec une face de découpe hexagonale



```

1 \begin{pspicture}(-4,-3)(4,5)
2 \psset{SphericalCoor=true,viewpoint=100
3   30 20,Decran=170}
4 \lightsource
5 \pstVerb{/distance 1.732 def
6   /n_x 0.57735 def
7   /n_y 0.57735 def
8   /n_z 0.57735 def
9   /xH distance n_x mul def
10  /yH distance n_y mul def
11  /zH distance n_z mul def
12  }%
13 \psframe(-4,-3)(4,5)
14 \psset{solidmemory}
15 % placement du cube dans le trièdre Oxyz
16 \psSolid[object=cube,a=2,
17   plansepare={
18     [ n_x % a
19       n_y % b
20       n_z % c
21       distance neg % d
22     ]},
23   name=parts_cube,
24   action=none](a 2 div,a 2 div,a
25     2 div)
26 \psSolid[object=load,
27   load=parts_cube1,
28   action=writesolid,
29   file=cubeHexagone]
30 \psSolid[object=datfile,
31   fcol=0 (Dandelion),
32   fillcolor={[rgb]{0.7 1 0.7}},
33   file=cubeHexagone]
34 \Normale[linecolor=red](distance
35   ,45,35.2644)
36 \psSolid[object=cube,a=2,action=draw
37   ](1,1,1)
38 \psProjection[object=texte,
39   fontsize=10,pos=dc,text=H,
40   normal=n_x n_y n_z -30](xH,yH,zH)
41 \axesIIID[linecolor=blue](2,2,2)
42 (2.5,2.5,2.5)
43 \end{pspicture}

```

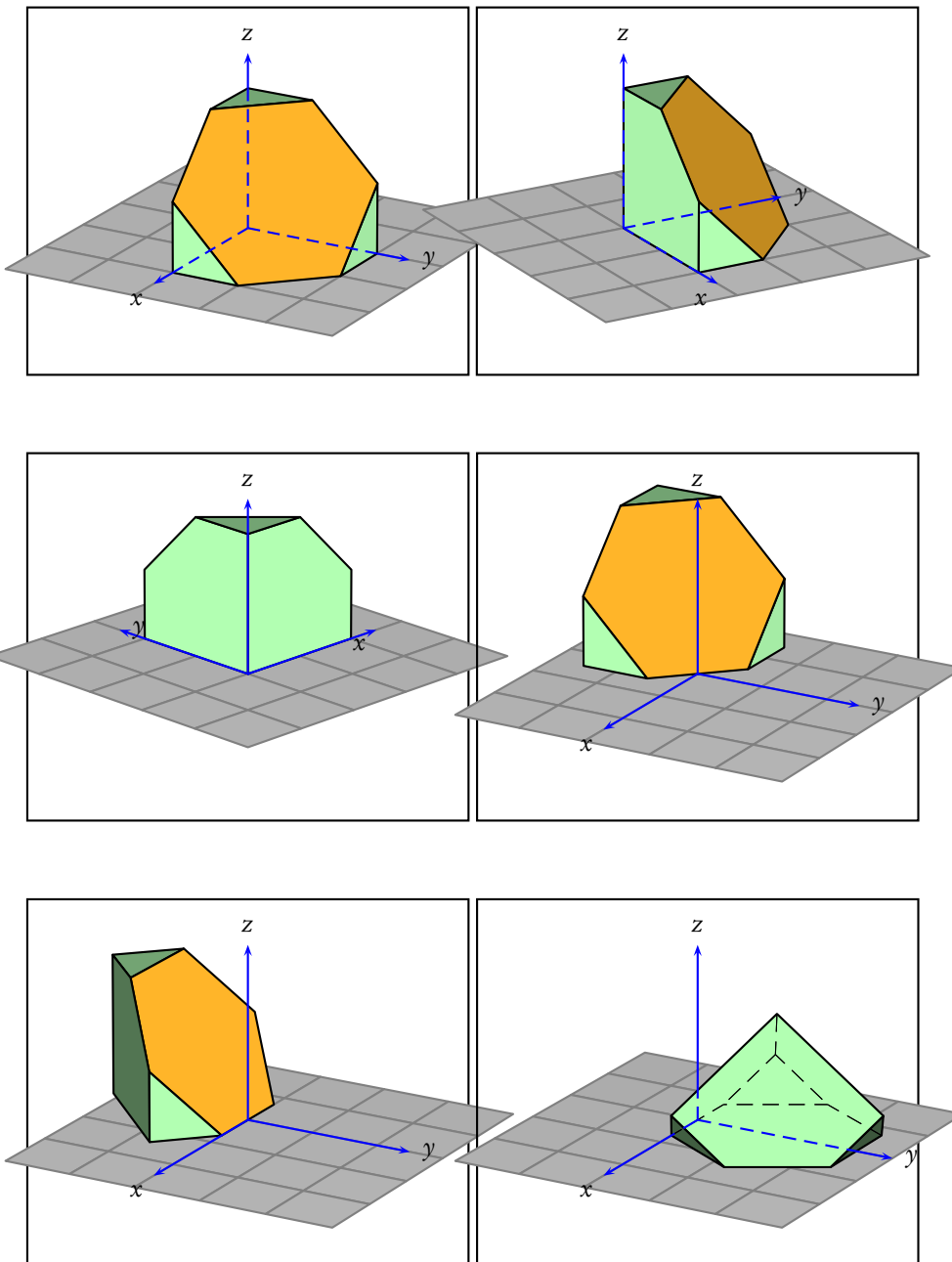
5.3 Cube découpé dans des différentes positions

Où on utilise l'option permettant de mémoriser un solide pour, après diverses transformations, faire reposer le cube tronqué sur sa face de découpe.

```

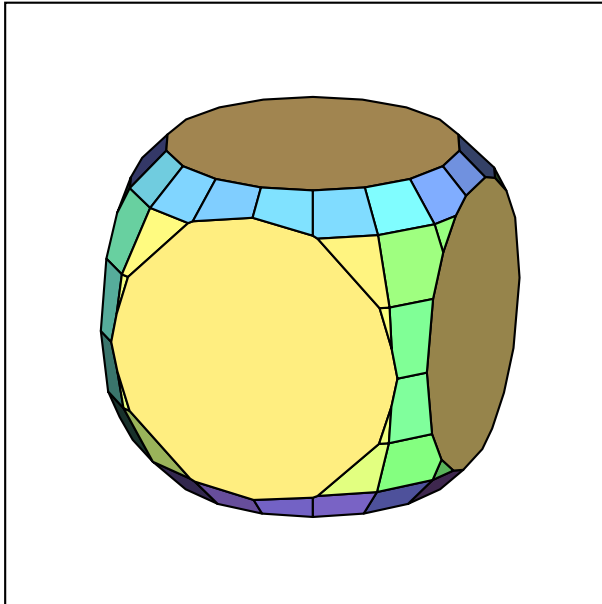
\psset{solidmemory}
\psSolid[object=datfile,
  fcol=0 (Dandelion),
  fillcolor={[rgb]{0.7 1 0.7}},
  name=C1,
  action=none,
  file=cubeHexagone]

```



6 Sections multiples

6.1 Coupes dans une sphère avec PStricks



```

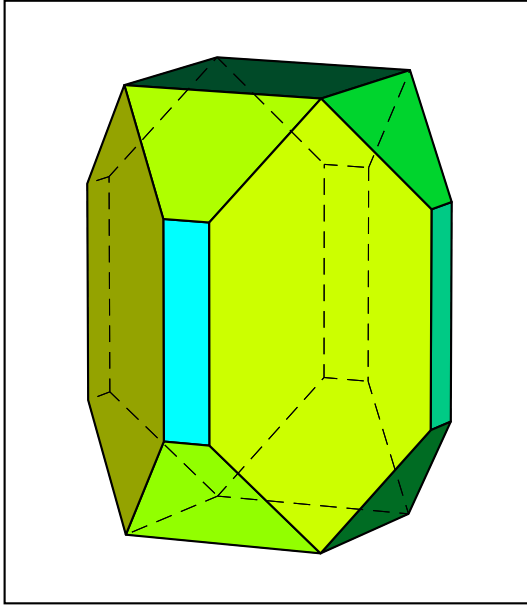
1 \psset{SphericalCoor=true,viewpoint=100
2   20 20,Decran=75}
3 \lightsource
4 \pstVerb{/coeff 0.75 def /r0 4 def /OH
5   coeff r0 mul neg def}%
6 \psset{solidmemory}
7 \begin{pspicture}(-4,-4)(4,4)
8 \psframe(-4,-4)(4,4)
9 \psSolid[object=sphere,
10  r=r0,
11  ngrid=9 18,
12  plansepare={[1 0 0 OH]},
13  name=part,
14  action=none]
15 \psSolid[object=load,
16  load=part1,plansepare={[-1 0 0 OH]},
17  action=none,name=part]
18 \psSolid[object=load,
19  load=part1,plansepare={[0 1 0 OH]},
20  action=none,name=part]
21 \psSolid[object=load,
22  load=part1,plansepare={[0 -1 0 OH]},
23  action=none,name=part]
24 \psSolid[object=load,
25  load=part1,plansepare={[0 0 1 OH]},
26  action=none,name=part]
27 \psSolid[object=load,
28  load=part1,plansepare={[0 0 -1 OH]},
29  action=none,name=part]
30 \psSolid[object=load,hue=.1 .8 0.5 1,
31  load=part1](0,0,0)
32 \composeSolid
33 \end{pspicture}

```

6.2 Sections multiples d'un parallélépipède

Les sections multiples gagneront à être exécutées dans une boucle postscript, dans `\codejps`, c'est plus simple et plus rapide !

Dans cet exemple, le solide de départ est un parallélépipède. Troncatures des sommets et chanfreinages des arêtes sont réalisées avec des plans de coupe successifs, d'abord les sommets puis les arêtes.

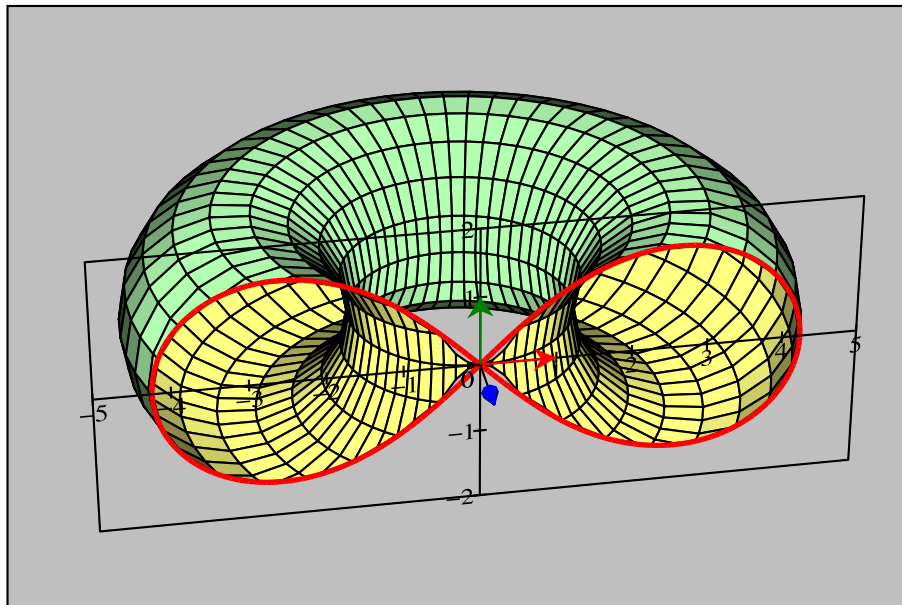
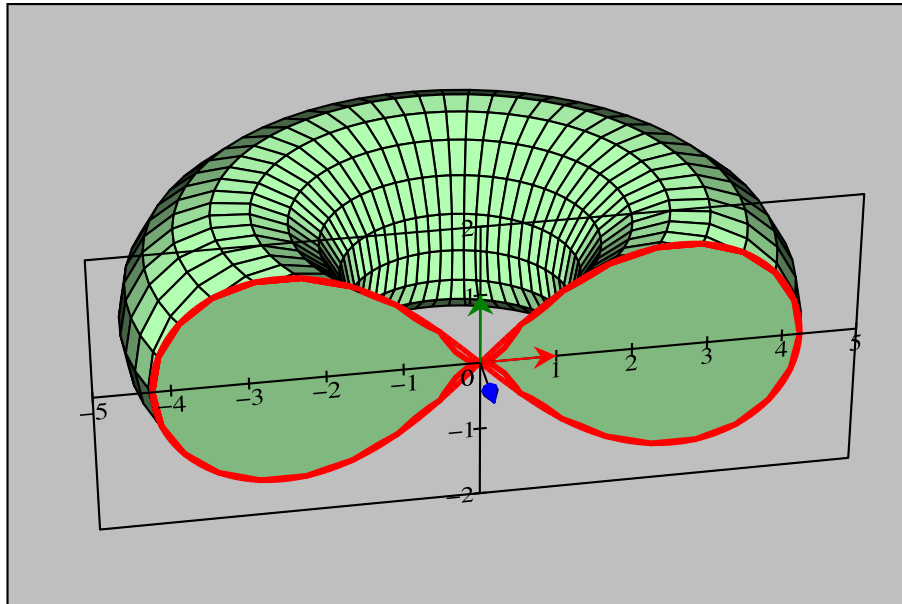


```

1 \begin{pspicture}(-3.5,-4)(3.5,4)
2 \psset{SphericalCoor=true,viewpoint=100 -20
3   10,Decran=100}
4 \lightsource
5 \psframe(-3.5,-4)(3.5,4)
6 \codejps{
7 4 4 6 newparallelepiped
8 45 90 360 {
9   /iAngle exch def
10   /n_x iAngle cos 35.2644 cos mul def
11   /n_y iAngle sin 35.2644 cos mul def
12   /n_z 35.2644 sin def
13   /distance 2 3 add 3 sqrt div neg def
14 [ n_x n_y n_z distance]
15 solidplansepare
16 } for
17 45 90 360 {
18   /iAngle exch def
19   /n_x iAngle cos 35.2644 cos mul def
20   /n_y iAngle sin 35.2644 cos mul def
21   /n_z 35.2644 sin neg def
22   /distance 2 3 add 3 sqrt div neg def
23 [ n_x n_y n_z distance]
24 solidplansepare
25 } for
26 45 90 360 {
27   /iAngle exch def
28   % plan : ax+by+cz-d=0
29   [ iAngle cos % a
30     iAngle sin % b
31     0 % c
32     -2.5 % -d
33 ] solidplansepare
34 } for
35 dup [.5 .2] solidputhuecolors
36 solidlightOn
37 drawsolid*}
38 \end{pspicture}

```


7 Sections d'un tore



8 Autres exemples

1. Vous trouverez une version codée de ce document en `jps` dans la commande `\codejps` dans le document suivant :
<http://melusine.eu.org/syracuse/mluque/solides3d2007/sections>
2. Une étude des sections coniques sur :

<http://melusine.eu.org/syracuse/mluque/solides3d2007/sections/sections-cone>

3. Une étude des sections cylindriques sur :

<http://melusine.eu.org/syracuse/mluque/solides3d2007/sections/section-cylindre>

4. Une étude sur les sections du tore :

<http://melusine.eu.org/syracuse/mluque/solides3d2007/sections/section-tore>

9 La commande `\lightsource`

La macro `lightsource` permet de placer la source de lumière au même endroit que le point de vue lorsque les coordonnées sphériques sont activées : `SphericalCoor=true`.

```
\def\lightsource{
\pstVerb{\pst@solides@viewpoint
/PhI exch def /ThetA exch def /Dist exch def
/Lx Dist ThetA cos mul PhI cos mul def
/Ly Dist ThetA sin mul PhI cos mul def
/Lz Dist PhI sin mul def}%
\psset{lightsrc=Lx Ly Lz}
}
```

Elle doit se placer après les coordonnées du point de vue.

```
\psset{SphericalCoor=true,viewpoint=50 -20 10,Decran=50}
\lightsource
```