pst-3d

Three dimensional views*

v.2.001

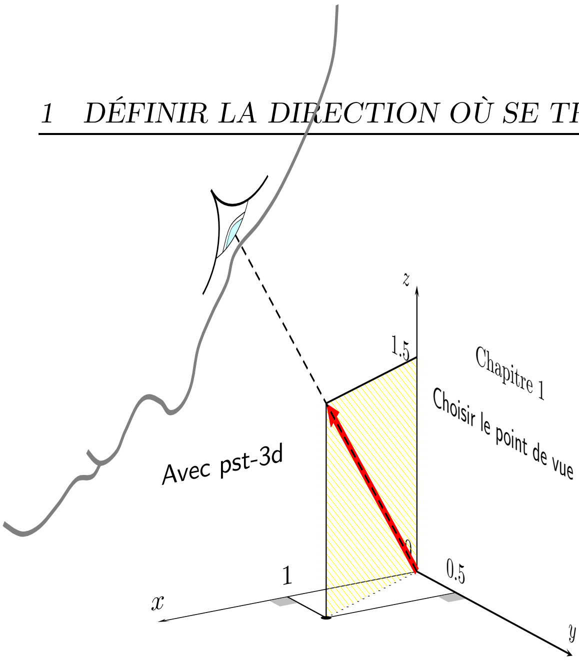
Manuel Luque and Herbert Voß

December 25, 2004

1 Définir la direction où se trouve l'observateur

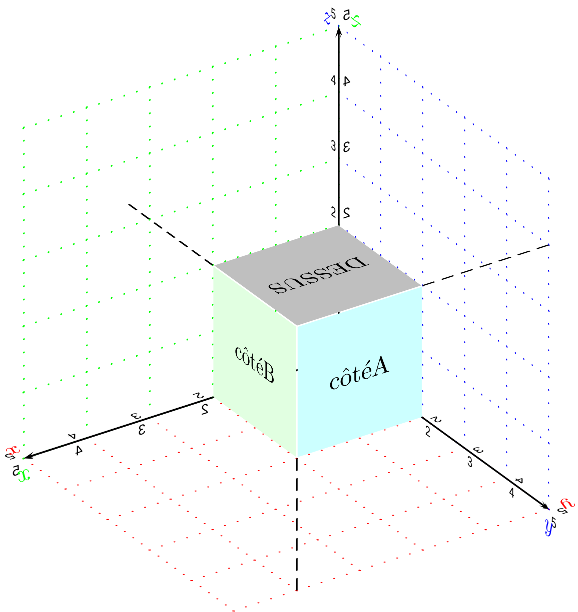
C'est le rôle de la commande `\psset{wiepoint=`$\mathtt{v_x\ v_y\ v_z}$`}`. ($\mathtt{v_x\ v_y\ v_z}$) sont les coordonnées d'un vecteur pointant à partir de l'origine O vers l'œil de l'observateur. En réalité il définit la direction de la projection sur le plan horizontal. La norme du vecteur n'a aucune importance, il est inutile de prendre un vecteur unitaire. Choisissez simplement les coordonnées en fonction du point de vue que vous voulez avoir par rapport à l'objet.
Dans le dessin ci-dessous le vecteur $\vec{v}$ a pour coordonnées (`1 0.5 1.5`). Ce schéma n'est pas représenté avec ce point de vue, car de ce vecteur nous ne verrions alors qu'un point : le bout de la flèche !
Remarque :*Il ne faut pas choisir l'une des coordonnées égale à 0, cela provoque une erreur lors de l'affichage de l'image, mais on peut prendre une valeur très petite : 0.001 par exemple. Il peut y avoir des erreurs provoquées par une division par 0 dans d'autres rares occasions, il suffit alors de modifier très légèrement la valeur fautive.*

*This document was written with `Kile: 1.7` (`Qt: 3.1.1; KDE: 3.3;` `http://sourceforge.net/projects/kile/`) and the PDF output was build with VTeX/Free (`http://www.micropress-inc.com/linux`)

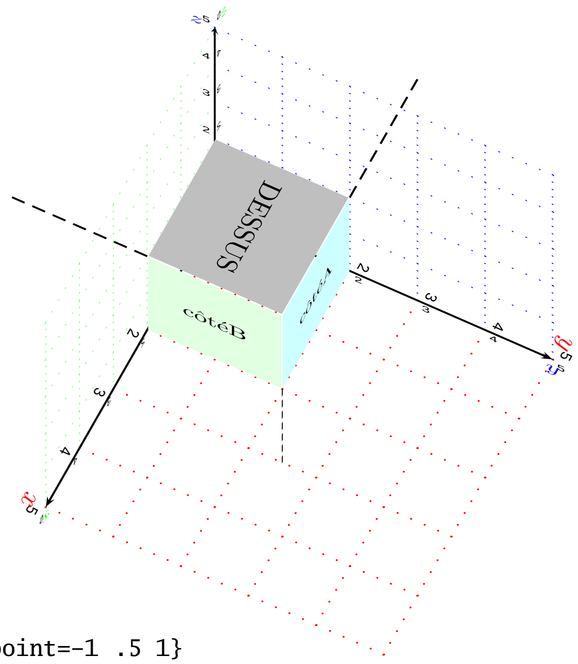


Trois exemples.

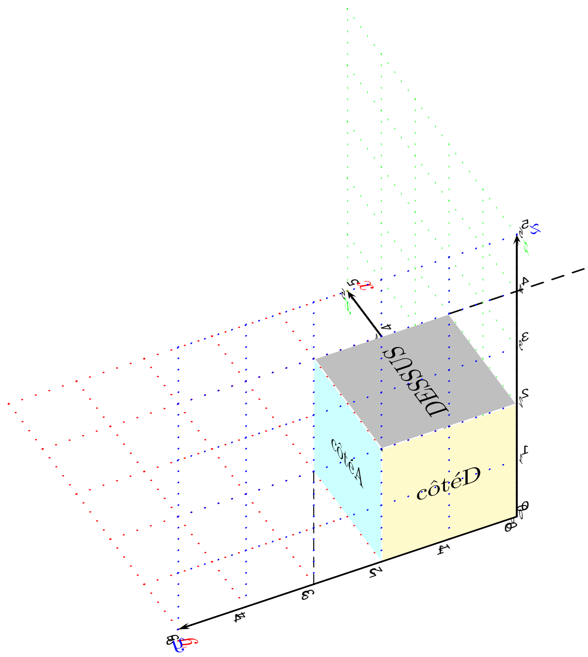
```
\psset{viewpoint=1 1.5 1}
```



```
\psset{viewpoint=1 .5 2}
```




```
\psset{viewpoint=-1 .5 1}
```



2 Choisir le plan sur lequel on veut travailler

Cette extension permet de tout représenter dans l'espace en 3d, à condition que ce que l'on souhaite visualiser appartienne à un plan. Si ce n'est pas le cas : cylindre, sphère etc. il faut découper la surface à représenter en pavés plans juxtaposés. En fonction de leur nombre, le rendu de la surface sera plus ou moins réaliste.

Définir le plan : nous disposons d'une commande (*qui est double en réalité*) très efficace : `\ThreeDput[normal=`n_x n_y n_z`](`Ω_x Ω_y Ω_z`)` qui permet à la fois :

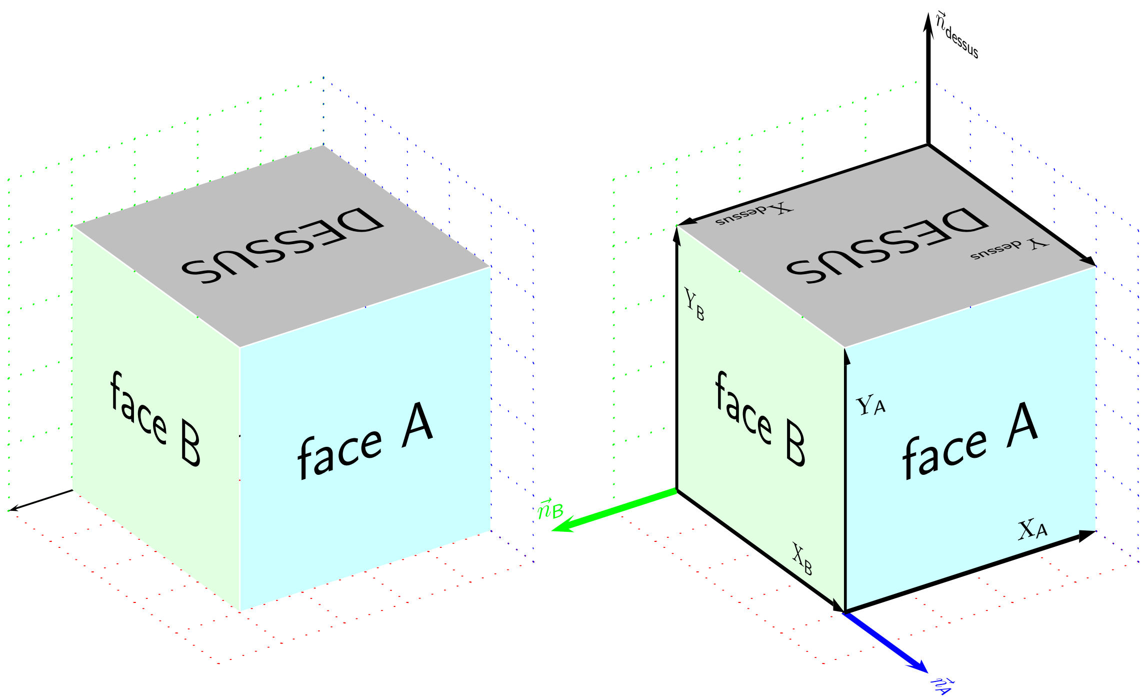
- de définir mathématiquement le plan par un vecteur normal à ce plan et par un point Ω appartenant à ce plan.
- Ce point Ω est considéré comme l'origine du repère du système d'axes $(\Omega\, X\, Y)$, sur lequel tous les repérages des points de ce plan se feront.

- Ce repère est automatiquement défini par la commande. Il suit les règles d'orientation classiques de l'espace que nous allons préciser par la suite.

Exemple 1 : représenter les faces visibles d'un cube

- face A : `\ThreeDput[normal=0 1 0](1,1,0)`
- face B : `\ThreeDput[normal=1 0 0](1,0,0)`
- face de dessus :`\ThreeDput[normal=0 0 1](0,0,1)`

Voici le cube que l'on a représenté avec ces paramètres et `\psset{wiepoint=1 1.5 1}` ainsi que la justification des paramètres choisis.



Dans la deuxième figure où sont représentés les vecteurs normaux aux trois faces ainsi que les repères associés, vous noterez la continuité dans l'orientation des axes si vous tournez autour du cube. De même faites pivoter le repère de la face de dessus sur lui-même, de façon à amener l'axe X_{dessus} à être parallèle à celui de la face B, basculez sur la face B, faites le glisser et vous verrez qu'il coïncide avec celui de la face B.

Quelles sont les règles d'orientation ?

Par exemple pour la face B :
Ω étant le point de cette face qui dans le repère de référence $Oxyz$ a pour coordonnées (1,0,0). $\Omega\,X_B$,$\Omega\,Y_B$,$\vec{n}_B$ forment un trièdre direct. Plus pratiquement avec la main droite, on doit avoir (les trois doigts formant un trièdre) :

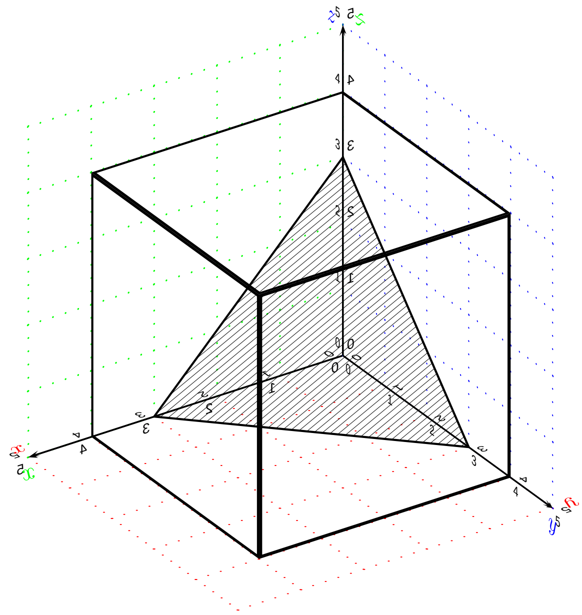
- pouce = $\Omega\,X_B$
- index = $\Omega\,Y_B$
- majeur = $\vec{n}_B$

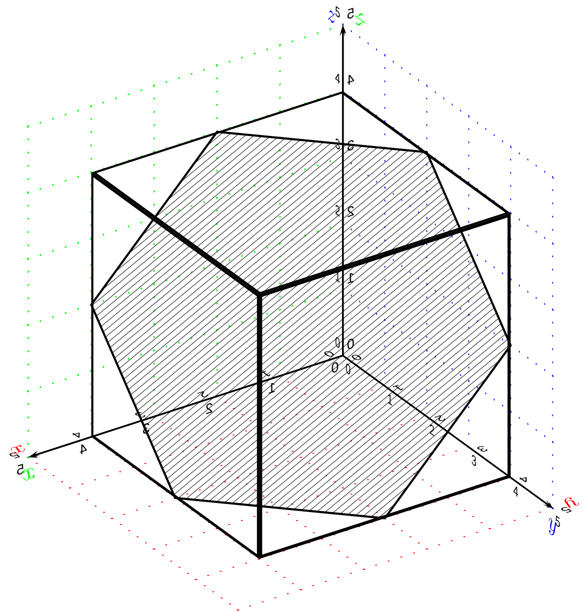
Toutes les faces suivent cette règle. C'est ainsi qu'en respectant cette règle, je peux écrire sur les faces du cube, de telle sorte qu'en restant à l'extérieur du cube je peux lire, en gardant les pieds sur le plan horizontal Oxy, les mots à l'endroit (*les normales sont orientées vers l'extérieur du cube*).

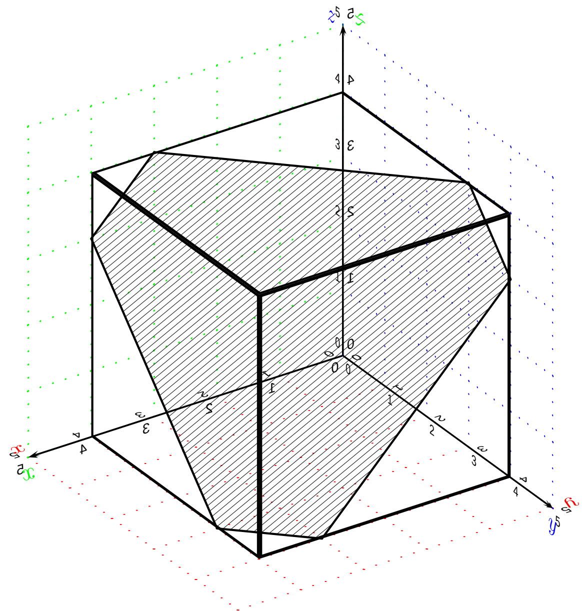
Étude de l'intersection du cube défini par :

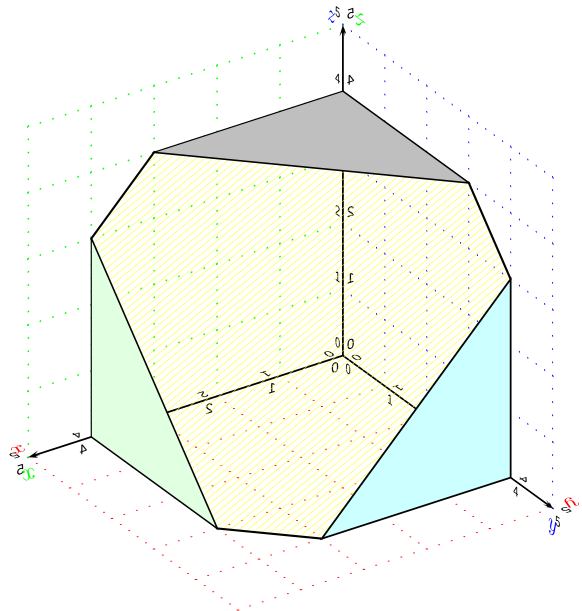
- $0 \leq x \leq 1$
- $0 \leq y \leq 1$
- $0 \leq z \leq 1$

Par le plan $x + y + z = h$ avec $0 < h < 3$
On définit ce plan par sa normale et l'origine Ω du nouveau repère dans ce plan :
Un plan d'équation cartésienne $ax+by+cz = d$ possède comme vecteur normal $\vec{n}(a, b, c)$. Dans le cas étudié, on peut donc prendre $\vec{n}(1, 1, 1)$, il est facile de vérifier que le point $\Omega(\frac{h}{3}, \frac{h}{3}, \frac{h}{3})$ est un point de ce plan, c'est le *pied* de la perpendiculaire de O à ce plan.
On définit parfaitement ce plan par la commande :`\ThreeDput[normal=1 1 1]`$(\frac{h}{3}, \frac{h}{3}, \frac{h}{3})$ Les exemples suivants sont relatifs respectivement à `h=0.75, h = 1.5 (hexagone régulier) et h = 1.75`.
Il faut calculer auparavant les coordonnées des sommets des polygones (triangle ou hexagone) intersections du plan avec les arêtes du cube (l'extension `pst-3d` ne le fait pas), puis dessiner le polygone par `\polygon....` ...









Par exemple, l'hexagone régulier est défini par :

```
\def\Rayon{0.707}%alpha=30°
\ThreeDput[normal=1 1 1](0.5,0.5,0.5){%(h/3,h/3,h/3)
    \SpecialCoor
    \pspolygon[fillstyle=hlines,hatchwidth=0.1pt,hatchsep=2pt]%
 (\Rayon;120)(\Rayon;180)(\Rayon;240)(\Rayon;300)(\Rayon;360)%
 (\Rayon;60)}
```

Les effets de transparence sont obtenus en hachurant l'hexagone avec du jaune (*rajouter à la liste des options* `hatchcolor=yellow`).