

tikz3d-fr

Quelques commandes (fr) pour
un peu de 3D avec TikZ.

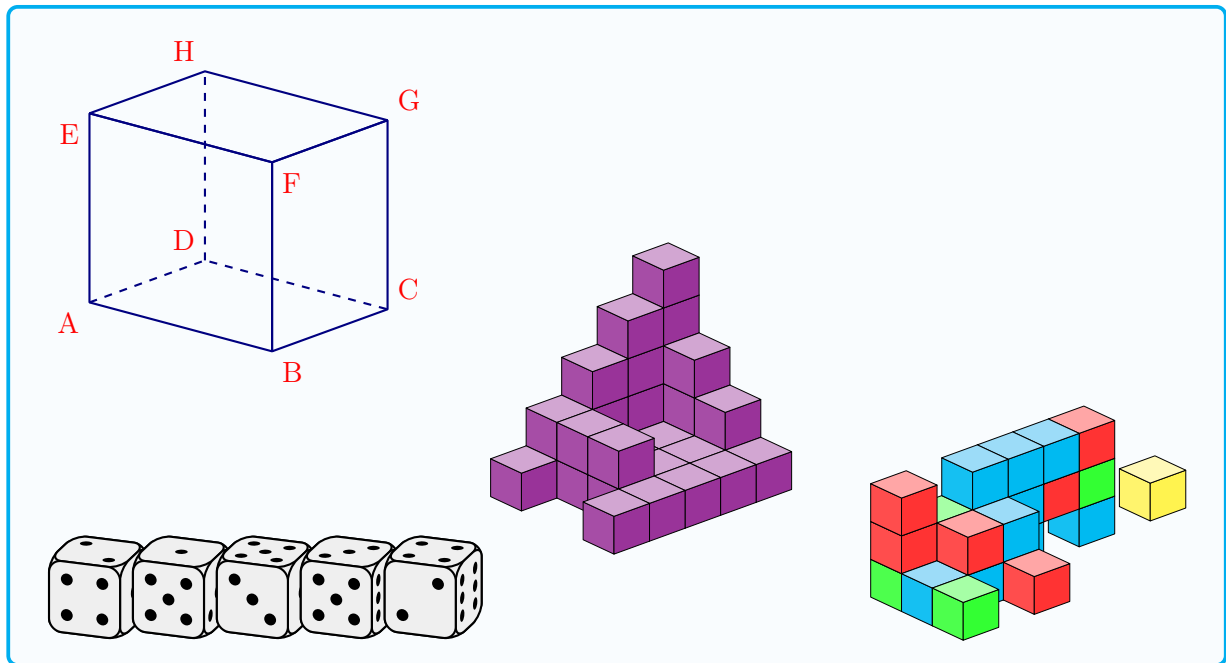
Version 0.1.4 - 5 mai 2025

Cédric Pierquet

c.pierquet - at - outlook . fr

<https://forge.apps.education.fr/pierquetcedric/packages-latex>

- ▶ Un environnement avec déclaration des unités.
- ▶ Une commande pour afficher un pavé, avec personnalisations.
- ▶ Deux commandes pour afficher des empilements de « petits cubes ».



L^AT_EX

pdfL^AT_EX

LuaL^AT_EX

TikZ

T_EXLive

MiK_TE_X

Table des matières

| | | |
|------------|--|-----------|
| I | Historique | 3 |
| II | Introduction | 4 |
| 1 | Le package tikz3d-fr | 4 |
| 1.1 | Introduction | 4 |
| 1.2 | Chargement du package, packages utilisés | 4 |
| III | Environnement 3D et commandes de base | 5 |
| 2 | Création de l’environnement | 5 |
| 2.1 | Commande | 5 |
| 2.2 | Clés et options | 5 |
| 3 | Points et segments | 7 |
| 3.1 | Commandes | 7 |
| 3.2 | Clés, options et arguments | 7 |
| 4 | Pavés | 9 |
| 4.1 | Commandes | 9 |
| 4.2 | Clés et options | 9 |
| IV | Empilements de cubes | 11 |
| 5 | Environnement dédié | 11 |
| 6 | Création par <i>plaques</i> | 11 |
| 6.1 | Commandes | 11 |
| 6.2 | Options et arguments | 12 |
| 7 | Création par <i>hauteurs</i> | 14 |
| 7.1 | Commande | 14 |
| 7.2 | Options et arguments | 14 |
| V | Dés à jouer, en 3D | 16 |
| 8 | Fonctionnement global | 16 |
| 9 | Influence des paramètres | 18 |
| 9.1 | Clés et options | 18 |
| 9.2 | Exemples | 18 |
| 9.3 | Angles de rotation | 19 |

Première partie

Historique

- v0.1.4 : Bugfix
- v0.1.3 : Affichage de dés en 3d (avec aléatoire possible)
- v0.1.2 : Option pour l'épaisseur des traits + `[line join=round]` pour les cubes
- v0.1.1 : Tracé des segments individuels avec l'option `[line cap=round]`
- v0.1.0 : Version initiale

Deuxième partie

Introduction

1 Le package tikz3d-fr

1.1 Introduction



Le package propose des commandes basiques – et francisées – pour travailler sur des figures simples en 3D, à l'aide de TikZ en utilisant des coordonnées tridimensionnelles :

- un environnement avec gestion des unités $x/y/z$;
- une commande pour tracer et personnaliser un cube ;
- des commandes pour créer/afficher/nommer des points de l'espace ;
- des commandes pour tracer un ou plusieurs segments ;
- des commandes et un environnement pour travailler sur des *empilements* de cubes ;
- des commandes pour représenter des dés en perspective (avec aléatoire possible).



Il existe d'autres solutions pour travailler avec de la 3D en L^AT_EX, comme par exemple les packages ProfCollege¹ (de Christophe Poulain, qui utilise MetaPost, et qui est certainement beaucoup plus performant) ou pst-ob3d² (de Herbert Voß et Denis Girou, qui utilise PSTricks). L'idée est de proposer une utilisation des capacités (natives) 3D de TikZ, en proposant des commandes *simplifiées* et *francisées* pour des figures simples (tétraèdres, cubes, pyramides, pavés) utilisées fréquemment dans des exercices de géométrie dans l'espace dans l'enseignement secondaires en France.

1.2 Chargement du package, packages utilisés



Le package se charge, de manière classique, dans le préambule. Il n'existe pas d'option pour le package, et xcolor n'est pas chargé.



</> Code L^AT_EX

```
\usepackage{tikz3d-fr}
```



tikz3d-fr charge les packages suivantes :

- tikz, xstring, xintexpr, simplekv et xinttools et listofitems, tikz-3dplot et randomlist ;
- la librairie *tikz.calc*.

Il est compatible avec les compilations usuelles en latex, pdflatex, lualatex ou xelatex.

1. <https://www.ctan.org/pkg/profcollege>

2. <https://www.ctan.org/pkg/pst-ob3d>

Troisième partie

Environnement 3D et commandes de base

2 Création de l'environnement

2.1 Commande



L'environnement dédié à la création de figures en 3D avec TikZ est `EnvTikzEspace`. Il permet de définir les unités et *angles* des différents axes.



```
</> Code LATEX
\begin{EnvTikzEspace}[Clés]<options tikz>
  %commandes
\end{EnvTikzEspace}
```

2.2 Clés et options



Le premier argument, optionnel et entre [...] propose les **(clés)** suivantes :

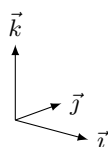
- **<UniteX>**, à donner sous la forme **<{angle:longueur}>** pour le vecteur de base \vec{i} ;
défaut : **<{-15:1cm}>**,
- **<UniteY>**, à donner sous la forme **<{angle:longueur}>** pour le vecteur de base \vec{j} ;
défaut : **<{20:0.65cm}>**
- **<UniteZ>**, à donner sous la forme **<{angle:longueur}>** pour le vecteur de base \vec{k} ;
défaut : **<{90:1cm}>**
- Le booléen **<VueClassique>** pour un affichage en perspective habituelle. défaut : **<false>**



Le second argument, optionnel et entre <...> est quant à lui relatif à des arguments à passer à l'environnement TikZ créé, comme par exemple un alignement vertical, etc



```
Code LATEX et sortie LATEX
\begin{EnvTikzEspace}
  \draw[->,>=latex] (0,0,0)--(1,0,0) node[right] {\vec{\imath}} ;
  \draw[->,>=latex] (0,0,0)--(0,1,0) node[right] {\vec{\jmath}} ;
  \draw[->,>=latex] (0,0,0)--(0,0,1) node[above] {\vec{k}} ;
\end{EnvTikzEspace}
```



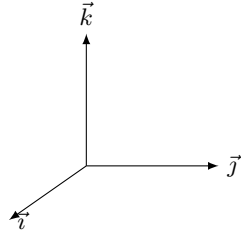


Code \LaTeX et sortie \LaTeX

```

\begin{EnvTikzEspace}[UniteX={-145:1.25cm},UniteY={0:1.75cm},UniteZ={90:1.75cm}]
  \draw[->,>=latex] (0,0,0)--(1,0,0) node[right] {$\vec{\imath}$} ;
  \draw[->,>=latex] (0,0,0)--(0,1,0) node[right] {$\vec{\jmath}$} ;
  \draw[->,>=latex] (0,0,0)--(0,0,1) node[above] {$\vec{k}$} ;
\end{EnvTikzEspace}

```

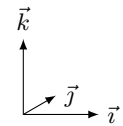


Code \LaTeX et sortie \LaTeX

```

\begin{EnvTikzEspace}[VueClassique]
  \draw[->,>=latex] (0,0,0)--(1,0,0) node[right] {$\vec{\imath}$} ;
  \draw[->,>=latex] (0,0,0)--(0,1,0) node[right] {$\vec{\jmath}$} ;
  \draw[->,>=latex] (0,0,0)--(0,0,1) node[above] {$\vec{k}$} ;
\end{EnvTikzEspace}

```



Une fois l'environnement, via son *repère*, est créé, toute commande en TikZ est utilisable avec les coordonnées (x,y,z) ou bien les *scope* avec les *canva*.

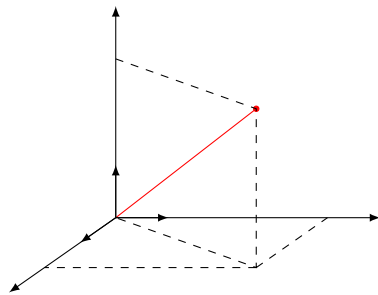


Code \LaTeX et sortie \LaTeX

```

\begin{EnvTikzEspace}[UniteX={-145:0.575cm},UniteY={0:0.7cm},UniteZ={90:0.7cm}]
  \filldraw[red] (2,4,3) circle[radius=1pt] ; \draw[red] (0,0,0) -- (2,4,3) ;
  \draw[thin,->,>=latex] (0,0,0)--(3,0,0) ;
  \draw[thin,->,>=latex] (0,0,0)--(0,5,0) ;
  \draw[thin,->,>=latex] (0,0,0)--(0,0,4) ;
  \draw[->,>=latex] (0,0,0)--(1,0,0) ;
  \draw[->,>=latex] (0,0,0)--(0,1,0) ;
  \draw[->,>=latex] (0,0,0)--(0,0,1) ;
  \draw[dashed] (0,0,3)--(2,4,3)--(2,4,0)--(0,0,0) (2,4,0)--(2,0,0) (2,4,0)--(0,4,0) ;
\end{EnvTikzEspace}

```



3 Points et segments

3.1 Commandes



Les commandes *simplifiées* et *francisées* disponibles sont :

- `\PlacePointEspace` pour placer un point dans l'espace ;
- `\PlacePointsEspace` pour placer des points dans l'espace ;
- `\MarquePointEspace` pour marquer (matérialiser) un point dans l'espace ;
- `\MarquePointsEspace` pour marquer (matérialiser) un point dans l'espace ;
- `\TraceSegmentEspace` pour tracer un segment dans l'espace ;
- `\TraceSegmentsEspace` pour tracer des segments dans l'espace.



</> Code \LaTeX

```
\begin{EnvTikzEspace}[Clés]<options tikz>
  %créer/placer/nommer un point
  \PlacePointEspace(*)[clés]{nœud}{coordonnées}<label>
  %créer/placer/nommer plusieurs points
  \PlacePointsEspace(*)[clés]{liste}
  %marquer un point
  \MarquePointEspace[clés]{point}
  %marquer plusieurs points
  \MarquePointsEspace[clés]{liste}
  %tracer un segment
  \TraceSegmentEspace[clés](point)(point)
  %tracer plusieurs segments
  \TraceSegmentsEspace[clés]{liste}
\end{EnvTikzEspace}
```

3.2 Clés, options et arguments



Les versions étoilées désactivent l'affichage des labels des points.

L'argument optionnel et entre [...] propose les **<clés>** suivantes (communes ou spécifiques) :

- **<PosLabel>** pour préciser la position (francisée) du label pour les points ; défaut : ****
- **<StyleMarque>** parmi **<x/o>** pour spécifier le style de la marque des points ;
défaut : **<o>**
- **<TailleMarque>** pour spécifier la taille de la marque des points (disque ou croix) ;
défaut : **<2pt>**
- **<Couleur>** pour paramétrer la couleur. défaut : **<black>**



Les positions pour les labels des points sont *francisées* :

- **** : bas
- **<h>** : haut
- **<g>** : gauche
- **<d>** : droite
- **<hg>** : haut gauche
- ...



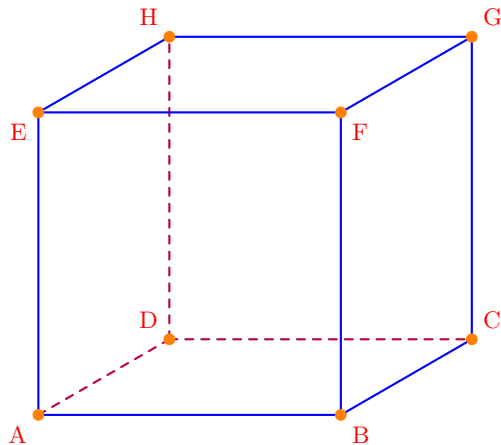
De manière un peu plus spécifique :

- le `<label>` pour la commande `\PlacePointEspace` est optionnel, et est identique à `{nœud}` ;
- la `{liste}` pour la commande `\PlacePointsEspace` est à donner – par exemple – sous la forme `A/0,0,0/bg B/5,2,1/hd` ;
- la `{liste}` pour la commande `\TraceSegmentsEspace` est à donner – par exemple – sous la forme `A/B A/C A/D B/D` ;
- les `[clés]` pour les segments correspondent aux options en langage `TikZ`.



Code \LaTeX et sortie \LaTeX

```
\begin{EnvTikzEspace}[VueClassique]
  %placement des points avec labels
  \PlacePointsEspace[Couleur=red]{A/0,0,0/bg B/4,0,0/bd C/4,4,0/hd D/0,4,0/hg
  ↪ E/0,0,4/bg F/4,0,4/bd G/4,4,4/hd H/0,4,4/hg}
  %segments pointillés
  \TraceSegmentsEspace[thick,dashed,purple]{A/D D/C D/H}
  %segments pleins
  \TraceSegmentsEspace[thick,blue]{A/B B/C C/G G/H H/E E/A E/F B/F F/G}
  %Marques points
  \MarquePointsEspace[Couleur=orange]{A,B,C,D,E,F,G,H}
\end{EnvTikzEspace}
```



4 Pavés

4.1 Commandes



La commande *simplifiée et francisée* pour afficher un pavé (ou un cube!) est la commande `\PaveTikzTriDim`.



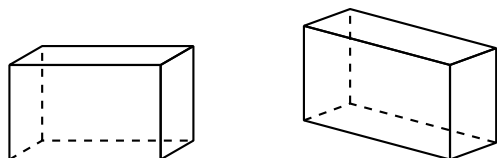
Code \LaTeX

```
\begin{EnvTikzEspace}[Clés]<options tikz>
  \PaveTikzTriDim[clés]
\end{EnvTikzEspace}
```



Code \LaTeX et sortie \LaTeX

```
\begin{EnvTikzEspace}[VueClassique]
  \PaveTikzTriDim
\end{EnvTikzEspace}
\hspace{1cm}
\begin{EnvTikzEspace}
  \PaveTikzTriDim
\end{EnvTikzEspace}
```



4.2 Clés et options



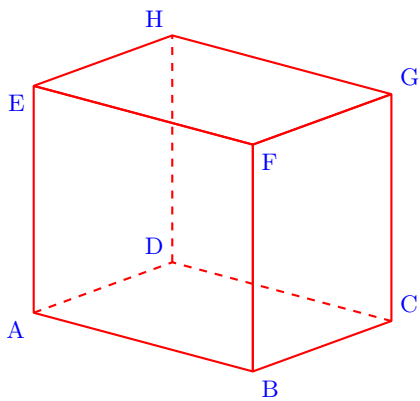
Quelques **clés** sont disponibles pour cette commande :

- **⟨Largeur⟩** : largeur du pavé; défaut **⟨2⟩**
- **⟨Profondeur⟩** : profondeur du pavé; défaut **⟨1⟩**
- **⟨Hauteur⟩** : hauteur du pavé; défaut **⟨1.25⟩**
- **⟨Sommets⟩** : liste des sommets (avec délimiteur §!); défaut **⟨A§B§C§D§E§F§G§H⟩**
- **⟨Math⟩** : booléen pour forcer le mode math des sommets; défaut **⟨false⟩**
- **⟨Epaisseur⟩** : épaisseur des arêtes (en *langage simplifié TikZ*); défaut **⟨thick⟩**
- **⟨AffLabel⟩** : booléen pour afficher les noms des sommets; défaut **⟨false⟩**
- **⟨Plein⟩** : booléen pour ne pas afficher les arêtes *invisibles*; défaut **⟨false⟩**
- **⟨Cube⟩** : booléen pour préciser qu'il s'agit d'un cube (seule **⟨Largeur⟩** est util(isé)e); défaut **⟨false⟩**
- **⟨Couleur⟩** : couleur des arêtes; défaut **⟨black⟩**
- **⟨CouleurSommets⟩** : couleur des sommets. défaut **⟨black⟩**



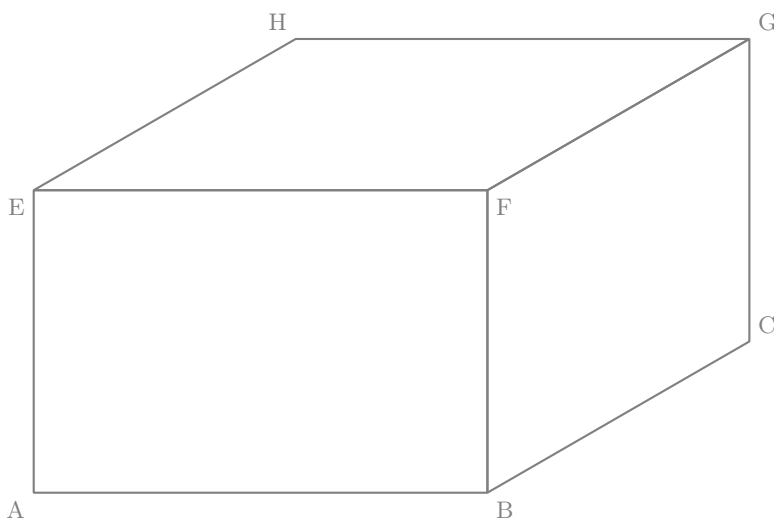
Code \LaTeX et sortie \LaTeX

```
\begin{EnvTikzEspace}  
  \PaveTikzTriDim[Cube,Largeur=3,Couleur=red,CouleurSommets=blue,AffLabel]  
\end{EnvTikzEspace}
```



Code \LaTeX et sortie \LaTeX

```
\begin{EnvTikzEspace}[VueClassique]<scale=2>  
  \PaveTikzTriDim[Largeur=3,Profondeur=4,Hauteur=2,Plein,Couleur=gray,AffLabel,  
  \to CouleurSommets=gray]  
\end{EnvTikzEspace}
```



Quatrième partie

Empilements de cubes

5 Environnement dédié



L'environnement dédié à la création de figures type *empilement de cubes* est... `EmpilementCubes`.

Il existe deux manières de définir les empilements :

- en travaillant par *plaques* verticales de l'arrière vers l'avant ;
- en travaillant par les *hauteurs* des colonnes, de l'arrière vers l'avant **et** de gauche à droite.

L'unité de base des cubes est fixée au départ à 0,5 cm.



Les axes (et de ce fait la vue proposée !) sont fixés, non modifiables, donc cette partie est beaucoup moins performante que ce propose le package `ProfCollege` avec sa commande `\VueCubes` !



```
</> Code  $\text{\LaTeX}$   
\begin{EmpilementCubes}[échelle]<options tikz>  
  %commandes  
\end{EmpilementCubes}
```

6 Création par *plaques*

6.1 Commandes



La commande pour créer une plaque *verticale* est `\PlaquePetitsCubes`, avec la contrainte de créer la *figure* de l'arrière vers l'avant.

Il existe également la commande `\PlaqueVide` pour *passer* une ligne.



```
</> Code  $\text{\LaTeX}$   
\begin{EmpilementCubes}[échelle]<options tikz>  
  \PlaquePetitsCubes[couleur(s)]{empilement}[épaisseur traits]  
  \PlaqueVide[nb]  
\end{EmpilementCubes}
```



Les plaques créées sont affichées l'une *devant* l'autre, et elles sont – par défaut – collées les unes aux autres.

6.2 Options et arguments



Le premier argument, optionnel et entre [...] permet de spécifier une couleur (cyan par défaut) ou une liste de couleurs qui seront utilisées pour la création des plaques :

- soit une couleur unique, qui sera *codée* par 1 pour la création des cubes ;
- soit plusieurs couleurs, sous la forme couleur1/couleur2/couleur3/... qui seront codées par 1, 2, ... pour la création des cubes.

Le second argument, obligatoire et entre {...} est quant à lui la liste, des lignes à construire, avec comme ordres :

- du bas vers le haut (caractère de séparation /) ;
- de la gauche vers la droite (caractère de séparation ,) ;
- un - code un *trou*, et un numéro code une couleur (comme définie(s) précédemment).

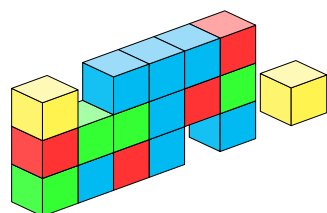
Le dernier argument, optionnel et entre [...] permet de spécifier une dimension pour les tracés (0.2pt par défaut).

Cette manière permet de créer des plaques avec couleurs *individuelles* et des *trous* éventuels.



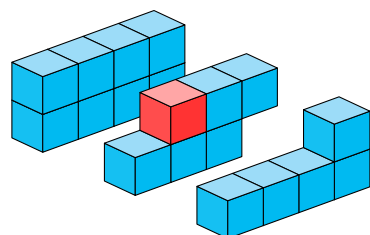
Code \LaTeX et sortie \LaTeX

```
\begin{EmpilementCubes}
  %plaque
  %de bas en haut : couleurs VBRB-B-J / RVVBRV / J-BBRR
  \PlaquePetitsCubes[cyan/red/green/yellow]{ 3121-1-4 / 233123 / 4-1112 }
\end{EmpilementCubes}
```



Code \LaTeX et sortie \LaTeX

```
\begin{EmpilementCubes}
  %plaque n°1 (fond)
  \PlaquePetitsCubes{ 1111 / 1111 }[0.4pt]
  \PlaqueVide[2]
  %plaque n°2
  \PlaquePetitsCubes[cyan/red]{ 111- / -211 }[0.4pt]
  \PlaqueVide[2]
  %plaque n°3 (devant)
  \PlaquePetitsCubes{ 1111 / ---1 }[0.4pt]
\end{EmpilementCubes}
```



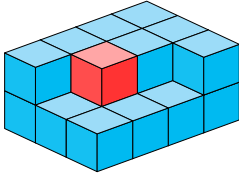


Code \LaTeX et sortie \LaTeX

```

\begin{EmpilementCubes}
  %plaque n°1 (fond)
  \PlaquePetitsCubes{ 1111 / 1111 }
  %plaque n°2
  \PlaquePetitsCubes[cyan/red]{ 111- / -211 }
  %plaque n°3 (devant)
  \PlaquePetitsCubes{ 1111 / ---1 }
\end{EmpilementCubes}

```

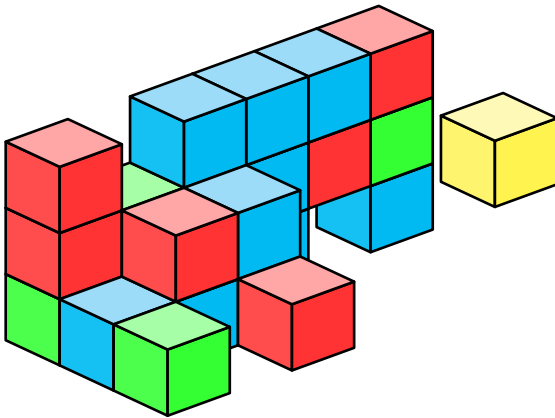


Code \LaTeX et sortie \LaTeX

```

\begin{EmpilementCubes}[1.75]
  \PlaquePetitsCubes[cyan/red/green/yellow]{ 3111-1-4 / 233123 / 2-1112 }[1pt]
  \PlaquePetitsCubes[cyan/red]{ 111 / -21 }[1pt]
  \PlaquePetitsCubes[cyan/red/green]{ 3-2 }[1pt]
\end{EmpilementCubes}

```



7 Création par hauteurs

7.1 Commande



L'idée, reprise du package ProfCollege³ permet d'afficher un empilement de cubes (monochromes, et sans trou) en précisant – grâce à un système de *grille* – les hauteurs des colonnes. La commande qui permet de réaliser cet empilement est `\BlocPetitsCubes`.



</> Code \LaTeX

```
%création dans un environnement dédié
\begin{EmpilementCubes}[échelle]<options tikz>
  \BlocPetitsCubes*[couleur]{grille des hauteurs}[epaisseur traits]
\end{EmpilementCubes}
```



</> Code \LaTeX

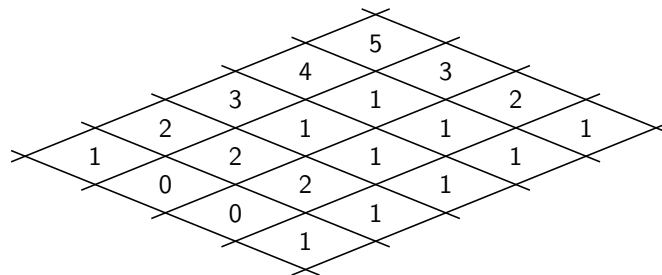
```
%création autonome
\BlocPetitsCubes[couleur]{grille des hauteurs}[epaisseur traits]
```



Le support de la grille des hauteurs est donc à donner sous forme *rectangulaire*, en respectant un nombre *homogène* de colonnes par ligne !



Voici une grille permettant d'anticiper la création d'un assemblage (en spécifiant les hauteurs) :



7.2 Options et arguments



Le premier argument, optionnel et entre [...] permet de spécifier une couleur (cyan par défaut).

Le second argument, obligatoire et entre {...} est quant à lui la liste des hauteurs, comme présentée précédemment :

- le caractère de séparation des *plaques* est le / ;
- pour chaque plaque, le caractère de séparation des colonnes est le , .

Le dernier argument, optionnel et entre [...] permet de spécifier une dimension pour les tracés (0.2pt par défaut).



Par exemple, la *grille* associée à l'empilement précédent est :

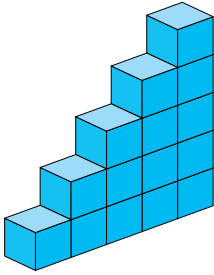
1,2,3,4,5 / 0,2,1,1,3 / 0,2,1,1,2 / 1,1,1,1,1

3. <https://www.ctan.org/pkg/profcollege>



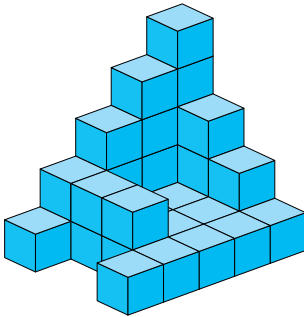
Code \LaTeX et sortie \LaTeX

```
\begin{EmpilementCubes}
  \BlocPetitsCubes*{1,2,3,4,5}
\end{EmpilementCubes}
```



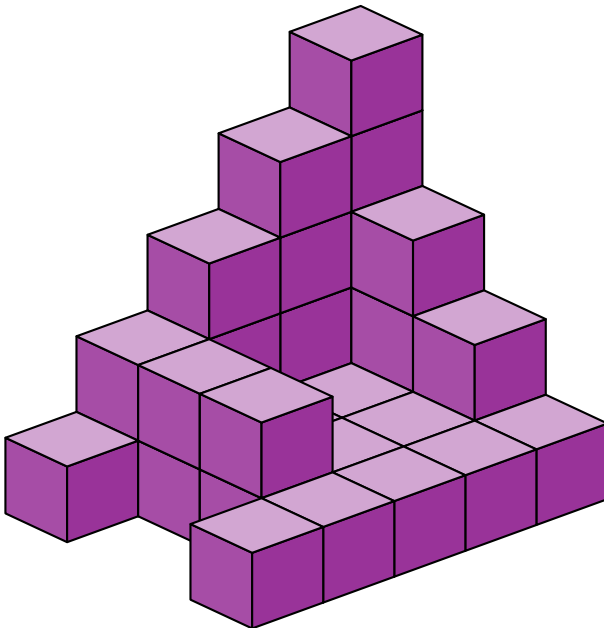
Code \LaTeX et sortie \LaTeX

```
%commande autonome, taille par défaut
\BlocPetitsCubes{1,2,3,4,5 / 0,2,1,1,3 / 0,2,1,1,2 / 1,1,1,1,1}
```



Code \LaTeX et sortie \LaTeX

```
\begin{EmpilementCubes}[2]
  \BlocPetitsCubes*[violet]{1,2,3,4,5 / 0,2,1,1,3 / 0,2,1,1,2 / 1,1,1,1,1}[0.8pt]
\end{EmpilementCubes}
```



Cinquième partie

Dés à jouer, en 3D

8 Fonctionnement global



L'idée est de proposer des commandes pour travailler avec des dés, en 3D, avec possibilité de les *lancer* de manière aléatoire.

La commande peut être autonome pour afficher dé par dé, mais elle peut être également à intégrer dans un environnement `tikzpicture`.



Par défaut, les dés ont une hauteur proche de 1 cm (sans la perspective).

L'orientation est *globalement* fixée, mais il est possible (dans une certaine mesure) de pivoter les dés (suivants les angles θ et ϕ des coordonnées sphériques).

Il est également possible de spécifier un *angle de vue*, droite ou gauche.



</> Code \LaTeX

```
%affichage indépendant  
\DeTriDim[clés]{numéro(s)}
```



</> Code \LaTeX

```
%affichage dans un environnement  
\begin{tikzpicture}  
  \DeTriDim*[clés]{numéro(s)}  
\end{tikzpicture}
```



</> Code \LaTeX

```
%tirages aléatoires  
\TirageAleaDesTriDim[clés]{tirages}
```



🔗 Code \LaTeX et sortie \LaTeX

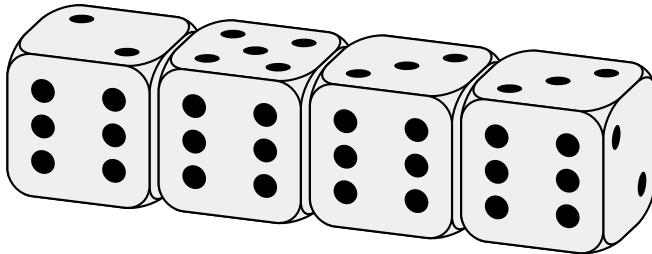
```
\DeTriDim{6/2}~\DeTriDim{1/5}~\DeTriDim{6/4}~\DeTriDim{3/2}
```





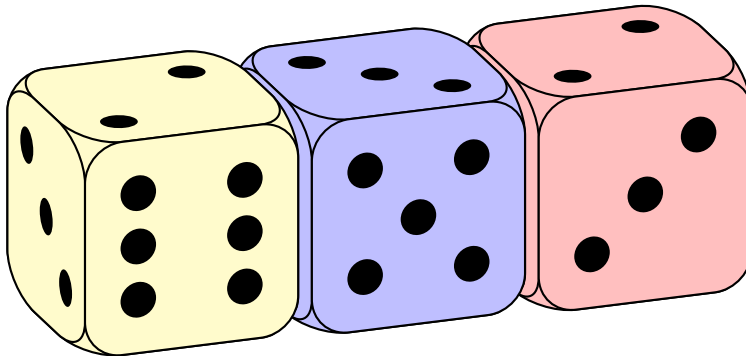
Code \LaTeX et sortie \LaTeX

```
\begin{tikzpicture}
  %\tdplotsetmaincoords{70}{110}
  \DeTriDim*[Echelle=2]{6/2}
  \DeTriDim*[Echelle=2,Origine={(2,-0.2)}]{6/5}
  \DeTriDim*[Echelle=2,Origine={(4,-0.4)},]{6/3}
  \DeTriDim*[Echelle=2,Origine={(6,-0.6)},Hasard]{}
\end{tikzpicture}
```



Code \LaTeX et sortie \LaTeX

```
\TirageAleaDesTriDim[Vue=G,Yams,Echelle=3,Couleurs={red!25,blue!25,yellow!25}]{3}
```



9 Influence des paramètres

9.1 Clés et options



Quelques **clés** sont disponibles pour cette commande (elles ne sont pas toutes utilisées par les toutes les commandes) :

- **⟨Echelle⟩** : échelle pour modifier la taille ; défaut **⟨1⟩**
- **⟨Couleur⟩** : couleur (individuelle) ; défaut **⟨lightgray!25⟩**
- **⟨CouleurPoints⟩** : couleurs des points ; défaut **⟨black⟩**
- **⟨Phi⟩** : angle ϕ de la perspective ; défaut **⟨110⟩**
- **⟨Theta⟩** : angle θ de la perspective ; défaut **⟨70⟩**
- **⟨Vue⟩** : vue de la perspective (D/G) ; défaut **⟨D⟩**
- **⟨Hasard⟩** : booléen pour un tirage aléatoire ; défaut **⟨false⟩**
- **⟨Origine⟩** : origine du dé pour la version `tikzpicture` ;
- en ce qui concerne les clés spécifiques au tirage :
 - **⟨EspaceH⟩** : espace horizontal entre les dés ; défaut **⟨0mm⟩**
 - **⟨Couleurs⟩** : liste des couleurs pour les dés ; défaut **⟨lightgray!25⟩**
 - **⟨Yams⟩** : booléen pour forcer un affichage en ligne. défaut **⟨false⟩**

Hormis dans le cas où `Harsard=true`, l'argument obligatoire, pour l'affichage unique, est à donner sous la forme `face de devant/face du dessus` (l'autre face visible est déterminée automatiquement).

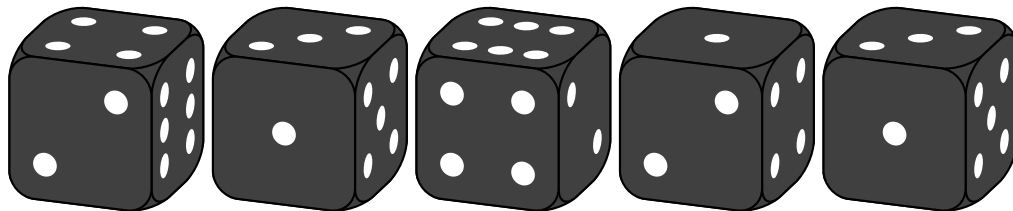
9.2 Exemples



Code \LaTeX et sortie \LaTeX

```
\TirageAleaDesTriDim[EspaceH=-2mm]{5}
```

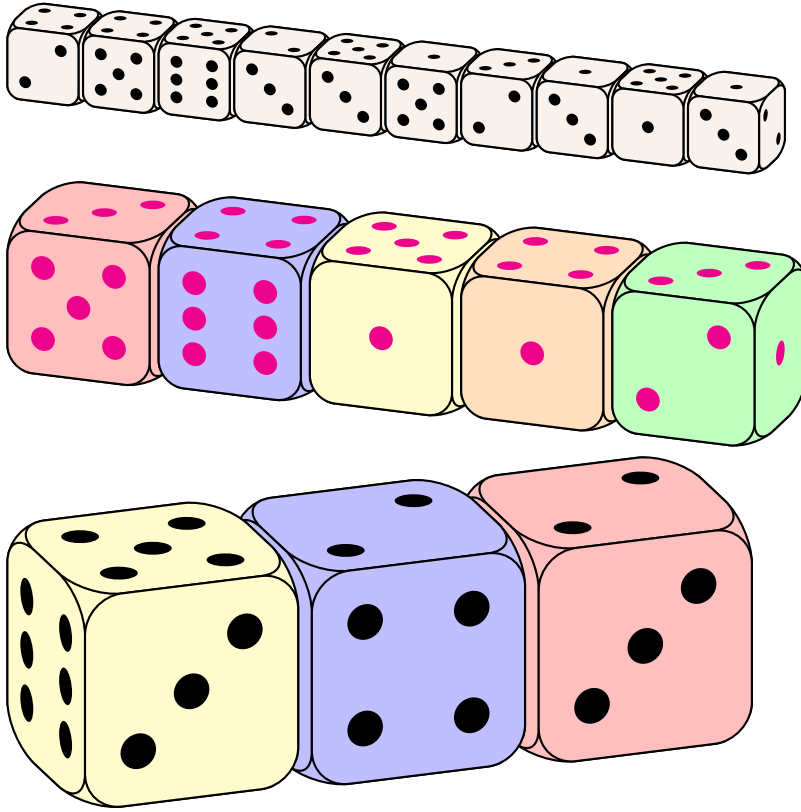
```
\TirageAleaDesTriDim[EspaceH=1mm,Echelle=2,Couleurs=darkgray,CouleurPoints=white]{5}
```





Code \LaTeX et sortie \LaTeX

```
\def\listecouleursdes{red!25,blue!25,yellow!25,orange!25,green!25}  
  
\TirageAleaDesTriDim[Yams,Couleurs=brown!10]{10}  
  
\TirageAleaDesTriDim[Yams,Echelle=2,Couleurs=\listecouleursdes,CouleurPoints=magenta]{5}  
  
\TirageAleaDesTriDim[Vue=G,Yams,Echelle=3,Couleurs=\listecouleursdes]{3}
```



9.3 Angles de rotation



Les paramètres θ et ϕ peuvent être modifiés, en fonction du rendu souhaité.

Mais comme toutes les faces ne sont pas traitées (du fait de la perspective), il est conseillé de rester dans une *fenêtre classique*.



Vue D :

| $\phi \backslash \theta$ | 60 | 70 | 80 | 85 |
|--------------------------|----|----|----|----|
| 95 | | | | |
| 100 | | | | |
| 110 | | | | |
| 120 | | | | |



Vue G :

| $\phi \backslash \theta$ | 60 | 70 | 80 | 85 |
|--------------------------|----|----|----|----|
| 95 | | | | |
| 100 | | | | |
| 110 | | | | |
| 120 | | | | |