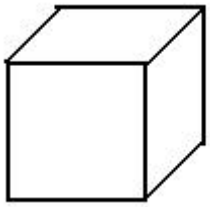
Géométrie en trois dimensions

Il s'agit de visualiser des objets en trois dimensions sur un plan, pour nous l'écran de l'ordinateur. Pour ce faire, nous allons simplifier les choses au maximum.

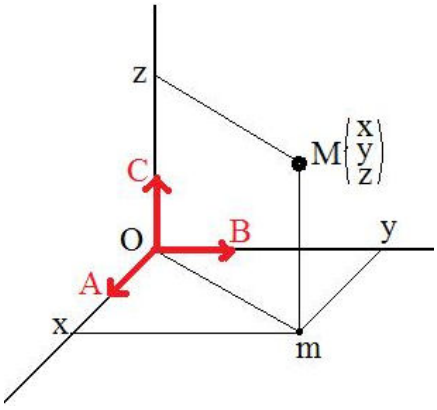
- Nous utilisons une perspective cavalière, à la façon d'un cavalier qui observe le paysage alentour du haut de son cheval. Ce type de perspective a comme propriété de respecter le parallélisme.



Par exemple, un cube est représenté comme sur le dessin ci-contre. Si ce type de perspective est intéressante par sa simplicité, elle ne permet pas d'obtenir l'effet de lointain, où les objets éloignés sont vus plus petits que ceux qui sont proches.

On passe de la scène que l'on observe en 3D à sa représentation dans un plan (tel un écran interposé entre l'œil et la scène) par une projection orthogonale. Cela signifie que les rayons issus de l'œil (en fait arrivant à l'œil) sont parallèles. Dans ces conditions, la distance de l'œil par rapport à la scène ne joue aucune rôle, pas plus que la position du plan de projection pourvu qu'il soit perpendiculaire aux rayons de l'œil. On sait que la projection respecte les proportions, et non pas les distances, ainsi que le parallélisme, notamment un carré devient un parallélogramme.

Le repère absolu dans lequel se trouve la scène est supposé orthonormé, et formé des trois vecteurs de base ***OA***, ***OB***, ***OC*** perpendiculaires deux à deux, et tous de longueur unité. Dans ce repère, les points du paysage sont connus par leurs coordonnées x, y, z.



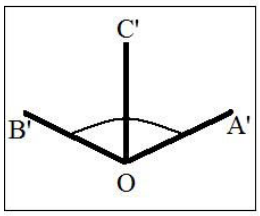
Le point M (x, y, z) se projette sur le sol - le plan OAB - en m qui a pour coordonnées x et y dans ce plan, et z est l'altitude du point M. On a vectoriellement :

$$\boldsymbol{OM} = \boldsymbol{Om} + \boldsymbol{mM} \quad \text{soit}$$
$$\boldsymbol{OM} = x\,\boldsymbol{OA} + y\,\boldsymbol{OB} + z\,\boldsymbol{OC}$$

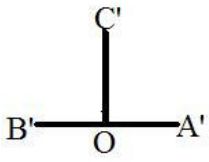
Remarquons que si nous projetons ces points sur un plan passant par O, avec M se projetant en M' , A en A', etc, on aura encore :
$\boldsymbol{OM'} = x\,\boldsymbol{OA'} + y\,\boldsymbol{OB'} + z\,\boldsymbol{OC'}$ puisque la projection orthogonale respecte les proportions.

- Simplifions encore : on considère que les rayons issus de l'œil sont parallèles au plan bissecteur des plans OAC et OBC. Dans ces conditions, le seul paramètre indiquant la position de l'œil est l'angle α entre la direction de l'œil et l'horizontale. L'œil voit le repère ***OA***, ***OB***, ***OC*** avec l'axe des z vertical, et les deux autres axes sont symétriques par rapport à l'axe vertical. Après projection orthogonale sur le plan écran passant par O, le repère devient ***OA'***, ***OB'***, ***OC'***. Il y a contraction des distances et modifications des angles auparavant droits. Ces variations sont toutes dépendantes de l'angle α. En faisant varier α entre 0 et $\pi/2$ (90°), on peut tourner autour de la scène, Au

prix de légers ajustements, on pourrait même faire un tour complet, avec α allant de 0° à 360° comme le fait un satellite tournant autour de la terre. Mais c'est le seul degré de liberté, à cause de nos conventions sur la position de l'œil. Pour avoir une totale liberté de manœuvre on pourrait ajouter une rotation de la scène autour de l'axe vertical *Oz*. Ces deux variations, l'une de l'œil, l'autre de la scène, suffiraient pour voir celle-ci sous tous les angles.

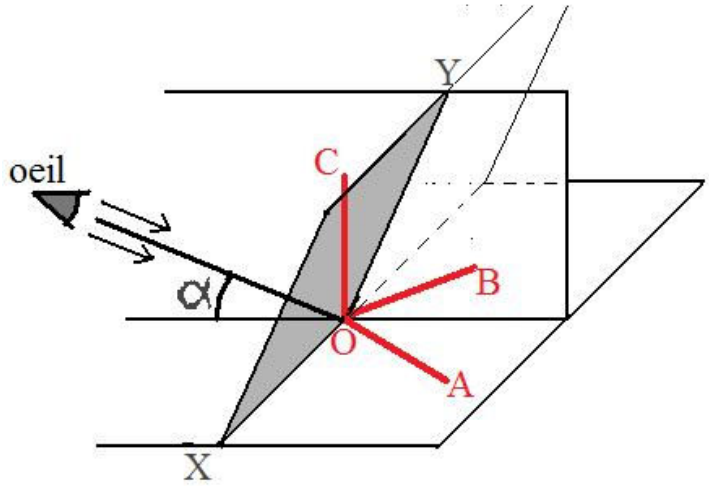


Le repère tel que le voit l'œil



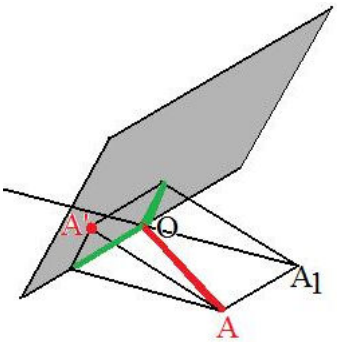
le repère pour α=0°

le repère pour α=90°



Vue d'ensemble : le repère *OA*, *OB*, *OC* dans lequel est placée la scène (le paysage, les objets 3D), la position de l'œil et ses rayons faisant un angle α avec l'horizontale, et le plan de projection en gris avec son repère orthonormé *OX*, *OY*

Relation entre l'inclinaison α de l'œil et la projection du repère sur l'écran



Passons au calcul. Le point *A* se projette en *A'* sur le plan de projection *OX*, *OY*. Comme ***OA*** fait un angle de 45° avec *OX*, on a $X_{A'} = 1 / \sqrt{2}$. De même $OA_1 = 1 / \sqrt{2}$. A son tour, A_1 se projette sur le plan de projection, et l'ordonnée du point projeté est $(1 / \sqrt{2})\sin \alpha$. D'où les coordonnées de *A'* dans le plan de projection :

$1/\sqrt{2}$, $(1/\sqrt{2})sin\alpha$.

Pour les mêmes raisons, celles de *B'* sont :

$-1/\sqrt{2}$, $(1/\sqrt{2})sin\alpha$.

Enfin le point *C* se projette en *C'*, et l'on obtient immédiatement ses coordonnées : 0, cosα.

Un point M (x, y, z) de l'espace se projette en M' et l'on a vu que :

OM' = x ***OA'*** + y ***OB'*** + z ***OC'***. Cela donne, par projection sur les axes OX, OY, les coordonnées de M' dans le plan de projection :

$X = x / \sqrt{2} - y / \sqrt{2} = (1/\sqrt{2})(x - y)$

$Y = x \sin\alpha / \sqrt{2} + y \sin\alpha / \sqrt{2} + z \cos\alpha = (\sin\alpha / \sqrt{2})(x + y) + z \cos\alpha$

Formules de passage de la 3D vers l'écran

Maintenant le plan de projection va devenir l'écran de l'ordinateur, à condition de pratiquer un zoom et de faire une translation des axes, avec le point O de coordonnées (xo, yo) sur l'écran. Le point M projeté sur l'écran a maintenant comme coordonnées

$xe = xo + A\,(x - y)$

$ye = yo - B(x + y) - Cz$

avec $A = zoom / \sqrt{2}$, $B = zoom \sin\alpha / \sqrt{2}$, $C = zoom \cos\alpha$

Ce sont les formules, finalement très simples, qui permettent de passer de l'espace 3D à sa visualisation sur l'écran.

Equation du plan de projection

En cas de besoin, déterminons l'équation du plan de projection, dans le repère OA, OB, OC. Ce plan étant perpendiculaire aux rayons issus de l'œil, il a pour vecteur normal (perpendiculaire à lui et de longueur 1) le vecteur ***N*** de coordonnées :

$\cos\alpha / \sqrt{2}$, $\cos\alpha / \sqrt{2}$, $- \sin\alpha$,

ce vecteur tant orienté dans le sens des rayons partant de l'œil. On en déduit que l'équation du plan est :

$(\cos\alpha / \sqrt{2})\, x + (\cos\alpha / \sqrt{2})\, y - \sin\alpha\, z = 0$, ou

$x + y - c\, z = 0$, avec $c = \sqrt{2} \tan\alpha$.

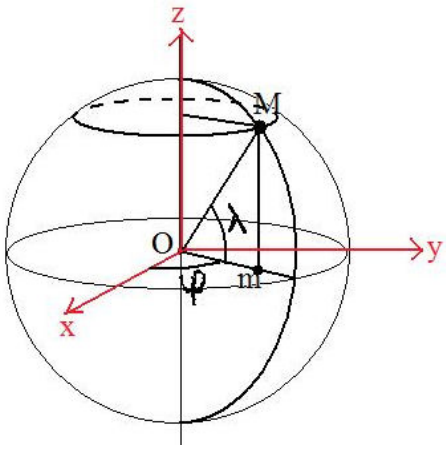
Tracé d'une sphère

Pour simplifier, nous supposons que le centre de la sphère est le point O, origine du repère en 3D. La sphère est définie par son centre et son rayon R. Un point M (x, y, z) appartient à la sphère si et seulement si $OM = R$, ce qui revient à $OM^2 = R^2$, et en appliquant le théorème de Pythagore, par analogie avec le cercle, on trouve :

$x^2 + y^2 + z^2 = R^2$.

Cette relation entre x, y et z est l'équation de la surface de la sphère. Mais cette formule n'est pas très pratique. On préfère déterminer la position d'un point M sur la sphère par le méridien et le parallèle sur lesquels il se trouve, ou encore par sa longitude et sa latitude. Les coordonnées d'un point M sur la sphère dépendent des deux angles φ –la longitude, et λ –la latitude, avec φ compris entre 0 et 2π, et λ entre $-\pi/2$ et $\pi/2$. D'où les équations paramétriques de la sphère, avec les coordonnées de M en fonction de φ et λ :

$x = R \cos\lambda \cos\varphi$
$y = R \cos\lambda \sin\varphi$
$z = R \sin\lambda$



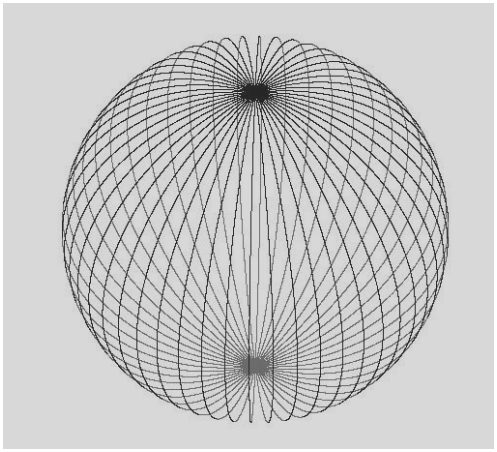
En se donnant la position de l'œil par l'angle α précédemment défini, on peut visualiser les méridiens (et si l'on veut les parallèles aussi) sur la sphère, en ne gardant que la partie visible. En fait cette partie visible est une demi-sphère, délimitée par le plan de projection d'équation $x + y - c\,z = 0$. Comme le vecteur normal $\boldsymbol{N}$ de ce plan est dirigé de l'avant vers l'arrière, on ne garde de la sphère que les points $M(x, y, z)$ tels que $x + y - cz < 0$.

Programme

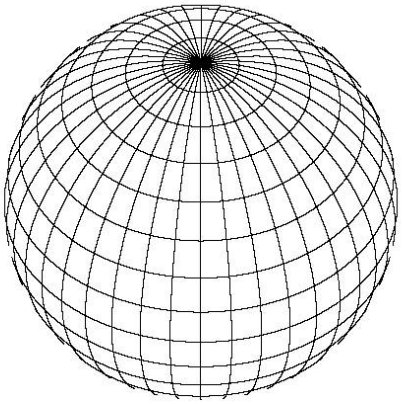
```
SDL_Init(SDL_INIT_VIDEO);
ecran=SDL_SetVideoMode(800,600,32, SDL_HWSURFACE|SDL_DOUBLEBUF);
blanc=SDL_MapRGB(ecran->format,255,255,255);
noir=SDL_MapRGB(ecran->format,0,0,0);
gris=SDL_MapRGB(ecran->format,100,100,100);
SDL_FillRect(ecran,0,blanc);

alpha=pi/4 ;  /* inclinaison de l'oeil donnée */
c=sqrt(2.)*tan(alpha);
A=zoom/sqrt(2.);  B=zoom*sin(alpha)/sqrt(2.);  C=zoom*cos(alpha);

for(phi=0.; phi<deuxpi; phi+=pi/25.)  /* on se contente ici des méridiens */
for(lambda=-pi/2.; lambda<pi/2.; lambda+=0.001)
{
  x=R*cos(lambda)*cos(phi); y=R*cos(lambda)*sin(phi);
  z=R*sin(lambda);
  xe=xorig +A*(x-y);   ye=yorig-B*(x+y)- C*z;
  if (x+y-c*z<0.) putpixel(xe,ye,noir);  /* partie visible en noir*/
  else if (getpixel(xe,ye)==blanc) putpixel(xe,ye,gris); /* partie cachée en gris */
}
SDL_Flip(ecran);
pause();
```



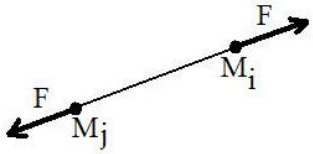
méridiens en noir devant
et derrière en gris



méridiens et parallèles

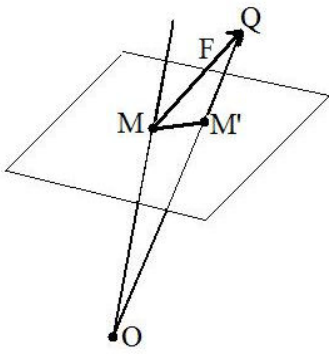
Exercice d'application : points se repoussant sur la sphère

On commence par placer N points M_i au hasard sur la sphère, numérotés de 0 à N-1, avec M_i de coordonnées ($px[i]$, $py[i]$, $pz[i]$). Ces coordonnées sont obtenues en prenant au hasard la longitude et la latitude de chaque point.



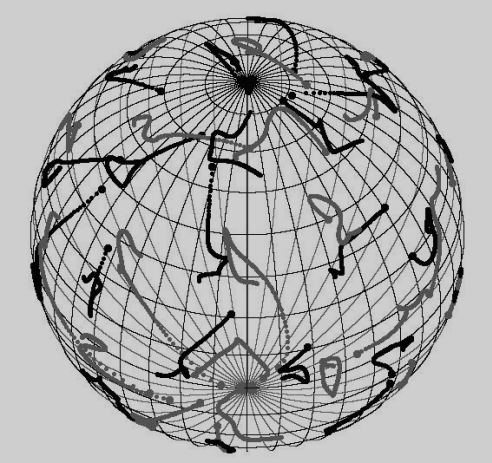
Ces points se repoussent deux à deux avec une force proportionnelle à la distance qui les sépare. Ils sont chacun soumis à la somme des forces de répulsion de tous les autres. Sous l'effet de ces forces, ils bougent sur la sphère.

Mais nous allons procéder ainsi : les points sont tous maintenus sur place, puis lâchés un court instant, puis retenus, puis relâchés, … Par ce procédé de petits mouvements répétés, les points vont atteindre une position finale d'équilibre, où plus aucun ne bouge. Chaque fois qu'on lâche un point pendant un court instant, son accélération est proportionnelle à la force de répulsion agissante ($\boldsymbol{F} = m\boldsymbol{A}$), et la vitesse passe de 0 à une faible vitesse $\boldsymbol{dV}$, avec $\boldsymbol{dV} = \boldsymbol{A}\, dt$ proportionnelle à l'accélération ou à la force. Seule la direction de $\boldsymbol{dV}$ nous intéresse, qui est aussi celle de la force, puisque le point bouge sur la sphère suivant la direction de $\boldsymbol{dV}$ (l'amplitude de la force ne nous concerne pas dans ce contexte). Il suffit de faire bouger le point très légèrement dans la direction de la force en le forçant à rester sur la sphère.

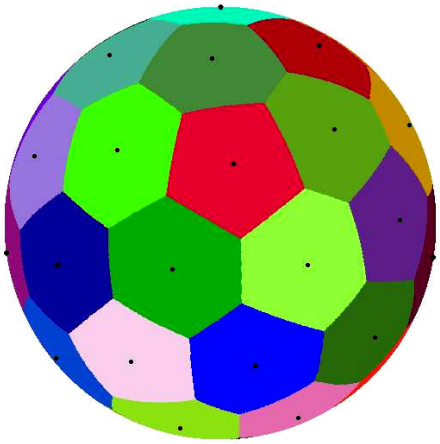


Méthode : pour chaque point M, lâché pendant un court instant puis bloqué, on calcule la force agissante, par cumul des forces provenant de la répulsion des autres points, auxquelles on donne une longueur très faible, d'où le vecteur $\boldsymbol{MQ}$. Seule la composante tangentielle, située dans le plan tangent à la sphère en M, joue. Il suffit de prendre le vecteur $\boldsymbol{OQ}$, et de déterminer le point M' situé sur (OQ) et dans le plan tangent, ou plus simplement de prendre M' sur (OQ) avec $OM' = R$ (R = 1 ici) afin de rester sur la sphère. Le point M' constitue la nouvelle position du point. Ensuite on recommencera à partir de M'. D'où le programme.

```
for(etape=1;etape <60000;etape++)   /* chaque étape de temps */
 {  kk=0.003;   /* longueur imposee à chaque force */
   for(i=0;i<N;i++) {fx[i]=0.; fy[i]=0.;fz[i]=0.;}
   for(i=0;i<N;i++)  /* pour chaque point i, on cumule les forces provenant des points j */
     { for(j=0;j<N;j++)  if (j!=i)
         {  r2=(px[j]-px[i])*(px[j]-px[i])+(py[j]-py[i])*(py[j]-py[i])
               +(pz[j]-pz[i])*(pz[j]-pz[i]); /* carré de la longueur de la force */
           fx[i]-=kk*(px[j]-px[i])/r2;fy[i]-=kk*(py[j]-py[i])/r2; fz[i]-=kk*(pz[j]-pz[i])/r2;
             /* cumul des petites forces de longueur kk */
         }
       qx=px[i]+fx[i];qy=py[i]+fy[i];qz=pz[i]+fz[i];   /* vecteur OQ */
       lq=sqrt(qx*qx+qy*qy+qz*qz); /* on va diviser ce vecteur par sa longueur lq */
       px[i]=qx/lq;py[i]=qy/lq;pz[i]=qz/lq; /* nouvelle position du point M  puisque R = 1 */
       xe=xorig +A*(px[i]-py[i]);    ye=yorig-B*(px[i]+py[i])- C*pz[i];
       if (px[i]+py[i]-c*pz[i]<0.) disque(xe,ye,1,noir);
       else disque(xe,ye,1,gris);
     }
  }
```



Déplacements des points vers leur position d'équilibre



Cellules de Voronoï des points

Au terme d'une série de petits mouvements (lâchers-blocages), les points atteignent une position d'équilibre. On peut enfin dessiner leurs cellules de Voronoï, c'est-à-dire prendre autour de chacun de ces points tous les points de la sphère qui sont plus près de lui que de tous les autres.