Annexe 2
Mathématiques et modélisation 3D sur Blender
Collège Albert Camus de Miramas

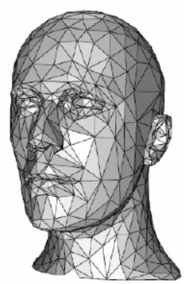
Nous sommes des élèves de quatrième et nous représentons les élèves qui ont travaillé sur le thème *mathématiques et modélisation 3D* de l'atelier Euclide du collège.

Le but de ces séances était à la fois de découvrir comment techniquement on produit des images de synthèse, mais aussi de travailler sur la place centrale que jouent les mathématiques dans ce domaine. On a travaillé sur *Blender*, un logiciel gratuit, mais très performant. Il est développé par des passionnés et il permet de modéliser tout type d'animation 3D.

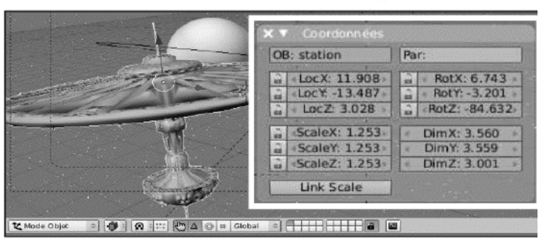
Le travail dans l'atelier a permis d'aborder les 4 compétences principales du logiciel :
- la modélisation (que l'on pourrait traduire par la fabrication d'objets),
- les textures (c.a.d. la manière d'habiller ces objets),
- l'animation (comment mettre ces objets en mouvement),
- la simulation de mécanismes physiques complexes, par exemple générer des particules physiques (faire du feu), gérer des collisions entre objets, ou bien encore l'écoulement d'un fluide.

En même temps que l'on a produit des séquences animées d'images de synthèse, on a exploré les opérations mathématiques qui se cachent derrière les principales fonctions du logiciels.

La modélisation



Modéliser un objet c'est en quelque sorte sculpter en 3D un volume en travaillant soit sur l'objet entier (on fait des rotations, on agrandit, ...) soit sur ses points : on déplace des points (ce que l'on appelle en maths une translation), on extrude, on symétrise, etc. En quatrième, on apprend justement à faire ces transformations avec la règle, le compas et le rapporteur : l'agrandissement (par la configuration de Thalès), la translation, la symétrie, la rotation … Mais *Blender* n'a pas de règle et d'équerre pour faire ses transformations. Comment fait-il, alors ? Le logiciel travaille en fait sur les coordonnées des points.



L'espace de *Blender* possède un repère qui permet de déterminer la position d'un objet dans l'espace grâce aux coordonnées x, y, z, de son centre de gravité et par la donnée de trois angles appelés angles d'Euler. Ces trois angles déterminent la manière dont l'objet est tourné par rapport à chaque axe.

Pour faire ces transformations, *Blender* calcule, à partir des coordonnées de départ, les coordonnées des points transformés. Pour cela, il utilise un outil mathématique puissant ; les **matrices**.

Une matrice, c'est quoi ? C'est un tableau de nombres disposés en lignes et colonnes qui représente une application linéaire. Par exemple une matrice 3×3 est un tableau de nombres de 3 lignes et 3 colonnes qui représente une application linéaire de $\mathbb{R}^3$: $\begin{pmatrix} 2 & 4 & 1 \\ 3 & 1 & -2 \\ 7 & 0 & 4 \end{pmatrix}$.

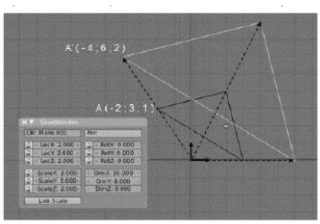
L'avantage en informatique, c'est qu'un tel tableau de nombres se programme facilement. Ces tableaux de nombres peuvent s'additionner et se multiplier :

$$\begin{pmatrix} 2 & 7 \\ 3 & 1 \end{pmatrix} + \begin{pmatrix} 6 & 4 \\ 5 & 8 \end{pmatrix} = \begin{pmatrix} 2+6 & 7+4 \\ 3+5 & 1+8 \end{pmatrix},$$

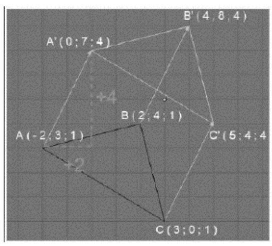
$$\begin{pmatrix} 2 & 7 \\ 3 & 1 \end{pmatrix} \times \begin{pmatrix} 6 & 4 \\ 5 & 8 \end{pmatrix} = \begin{pmatrix} 2\times 6+7\times 5 & 2\times 4+7\times 8 \\ 3\times 6+1\times 5 & 3\times 4+1\times 8 \end{pmatrix}.$$

Elle peuvent aussi, et ce qui nous intéresse ici, agir sur les coordonnées par multiplication.
Or, on va voir que les transformations mathématiques comme l'agrandissement ou la symétrie peuvent s'écrire sous la forme de matrices.
Notre travail a été de trouver la matrice qui opère à chaque transformation que l'on fait.

On a commencé par l'agrandissement à l'échelle 2 d'une figure simple comme ce triangle. On a observé que le vecteur OA (–2, 3, 1) est devenu OA′ (–4, 6, 2). Oui, mais quelle matrice a fait le travail pour obtenir OA′ à partir de OA ? Tout le monde a assez rapidement trouvé qu'il fallait des 2 sur la diagonale et des 0 partout ailleurs : $\begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}$



On a fait le même travail pour les différentes symétries et pour la rotation. Par contre la translation a posé un problème : On a appliqué à notre triangle une translation selon le vecteur (2, 4, 3). Chaque point a gagné 2 unités au niveau des abscisses, 4 unités au niveau des ordonnées et 3 en hauteur. Est-il possible trouver une matrice qui permette de réaliser cette transformation ? Eh bien, personne n'y arrivait. Cela tient au fait que la translation n'est pas une application linéaire de $\mathbb{R}^3$ dans lui-même : $t(kx) = kx + u \neq k\ t(x)$. Comment faire alors ?



Un espace de travail en 2D comme un plan peut être vu comme plongé dans un espace en 3D. L'idée pour la translation est de considérer l'espace à 3 dimensions plongé dans un espace à 4 dimensions en ajoutant une quatrième coordonnée factice. Il est alors simple de voir que la matrice à utiliser est : $\begin{pmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$.

Pour rendre l'ensemble du travail homogène, on a réécrit chaque matrice de transformation avec 4 lignes et 4 colonnes :

Pour un agrandissement de rapport k :

$$\begin{pmatrix} k & 0 & 0 & 0 \\ 0 & k & 0 & 0 \\ 0 & 0 & k & 0 \\ 0 & 0 & 0 & k \end{pmatrix}$$

Pour une symétrie par rapport au plan (XOY)

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Pour la rotation autour de l'axe (Ox) :

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Pour la translation de vecteur (t_x,t_y,t_z) :

$$\begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Tout ce travail nous a permis de vérifier par des calculs que les transformations qui permettent de modéliser les objets sont gérées par les mathématiques.

Les couleurs dans Blender

Pour les couleurs, on a découvert que *Blender* code ses couleurs en mode RVB, c'est-à-dire qu'il utilise 3 couleurs : le Rouge, le Vert et le Bleu. L'information est codée en binaire. Une mémoire élémentaire, que l'on appelle en informatique un bit, a 2 états possibles : 0 ou 1.

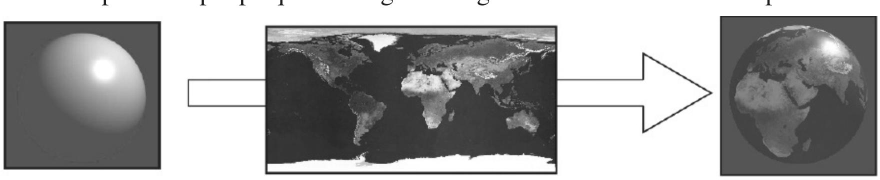
– À l'aide de 1 bit on a donc 2 possibilités 0/1 ou ouvert/fermé ou noir/blanc.

– À l'aide de 2 bits on a $2 \times 2 = 4$ possibilités : 00/01/10/11.

Blender code chacune de ses couleurs à l'aide de 8 bits ($2^8 = 256$ possibilités), ce qui offre 256 nuances. En convertissant ces nombres en système binaire avec des 0 et des 1 et en mettant ces trois nombres bout à bout, on obtient par exemple le code suivant pour une couleur « chair » : 111110111101000010010111 (Rouge 251 soit 11111011, Vert 208 soit 11010000, Bleu 151 soit 10010111)

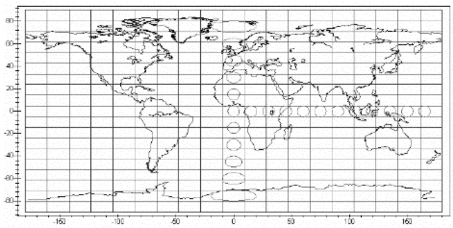
Plaquer une texture dans Blender

Comment par exemple plaquer l'image rectangulaire de la terre sur une sphère ?



Pour plaquer cette image sur une sphère, on donne l'ordre au logiciel que la sphère mathématique devienne un matériau. Sur ce matériau, on va plaquer une texture en appelant cette image. Puis on annonce au logiciel qu'il devra projeter cette image sur une sphère.

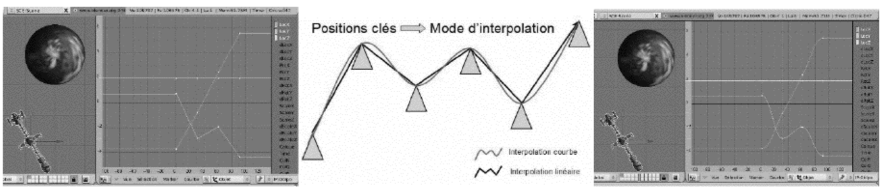
On peut se demander alors quel type de projection mathématique est utilisé lorsque l'on active le mode « sphère ». *Blender* utilise en fait la plus simple des projections cylindriques. Un point se repère par une longitude Ouest ou Est par rapport au méridien de Greenwich : de −180° à +180° et par une latitude sud ou nord : de −90° à +90°.



Lorsqu'on lui demande de plaquer cette image sur la sphère, Blender se contente de considérer que l'abscisse de chaque point de l'image correspond à la longitude, et l'ordonnée à la latitude.

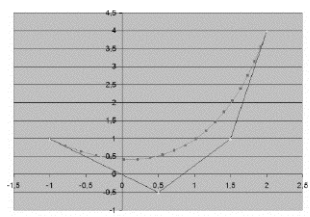
L'animation et les courbes de Bézier

Pour animer un objet, *Blender* réclame simplement de lui donner des positions clés espacées dans le temps. Dans *Blender*, ces positions clés offrent des valeurs que le logiciel utilise pour calculer alors toutes les positions intermédiaires (appelées dans *Blender* courbes « IPO ») par une méthode mathématique appelée « interpolation ». L'interpolation peut être courbe ou linéaire.



Au collège, on étudie les fonctions linéaires et affines : c'est ce qu'utilise Blender pour relier les positions clés par un déplacement rectiligne. Le problème de monsieur Bézier, qui était ingénieur dans l'automobile, était d'arriver à trouver une méthode mathématique qui permette de lisser par une courbe un ensemble de points reliés par des segments. Il a proposé d'utiliser pour cela deux fonctions qui permettent de calculer l'abscisse x et l'ordonnée y des points de cette courbe en fonction d'une variable t qui varie entre 0 et 1. Ces deux polynômes se construisent autour des coefficients du triangle de Pascal et autour des puissances de t et de $(1 - t)$.

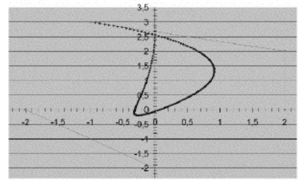
Pour faire une interpolation de Bézier sur 3 points, on utilise une courbe de degré 2 :



$$x(t) = x_A\left(1-t\right)^2 + 2x_Bt\left(1-t\right) + x_Ct^2,$$

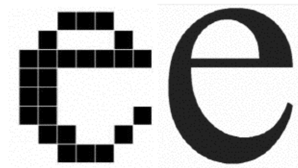
$$y(t) = y_A\left(1-t\right)^2 + 2y_Bt\left(1-t\right) + y_Ct^2.$$

Pour 4 points, on utilise une cubique :



$$x(t) = x_A(1-t)^3 + 3x_B t(1-t)^2 + 3x_C t^2(1-t) + x_D t^3,$$
$$y(t) = y_A(1-t)^3 + 3y_B t(1-t)^2 + 3y_C t^2(1-t) + y_D t^3.$$

Pour 5 points, une courbe de degré 4, etc.



Pour bien se rendre compte que cela fonctionne, on a fait de nombreux calculs à la main de coordonnées de points de courbes de Bézier. On a aussi appris à se servir d'Excel pour trouver les équations de Bézier qui permettent de produire des lettres de l'alphabet.
C'est une autre application de ces courbes : dans Blender, il est possible de mettre du texte, pour faire des titres. Comment sont faites les lettres par l'ordinateur ? Dans les années 80, l'ordinateur avait en mémoire un dessin de chacune des 26 lettres de l'alphabet. Une lettre était stockée sous la forme d'une grille de 8 lignes et 8 colonnes. Chaque case était noire ou blanche, ce qui en mémoire correspond au symbole 0 ou 1.
Mais cette méthode avait des inconvénients. Si l'on voulait grossir le texte, l'ordinateur grossissait la grille, et on voyait apparaître des gros carrés.

Avec les ordinateurs modernes, ces lettres sont obtenues par des courbes de Bézier et l'ordinateur recalcule des détails supplémentaires à chaque nouvel agrandissement

Pour finir, on a étudié comment géométriquement sont produits les points de la courbes de Bézier. Par exemple, dans une interpolation de Bézier sur 3 points A, B et C on cherche :
– le barycentre M des points A et B affectés des masses 1 − t et t : $(1-t)\overrightarrow{MA} + t\overrightarrow{MB} = 0$ donne $\overrightarrow{AM} = t\overrightarrow{AB}$;
– puis le barycentre N des points B et C affectés toujours des masses 1 − t et t : $\overrightarrow{BN} = t\overrightarrow{BC}$;
– enfin le point G, barycentre des points M et N affectés encore des masses 1 − t et t : $\overrightarrow{MG} = t\overrightarrow{MN}$. G est en final un point de la courbe. Il suffit alors faire varier t entre 0 et 1 et on trouve tous les points de la courbe. En exprimant le vecteur OG en fonction de OA, OB et OC grâce à la relation de Chasles, on a retrouvé nos courbes de Bézier :

$$\overrightarrow{MO} + \overrightarrow{OG} = t\overrightarrow{MO} + t\overrightarrow{ON}$$
$$\overrightarrow{OG} = (1-t)\overrightarrow{OM} + t\overrightarrow{ON}$$
$$\overrightarrow{OG} = (1-t)\overrightarrow{OA} + (1-t)\overrightarrow{AM} + t\overrightarrow{OB} + t\overrightarrow{BN}$$
$$\overrightarrow{OG} = (1-t)\overrightarrow{OA} + (1-t)t\overrightarrow{AB} + t\overrightarrow{OB} + t^2\overrightarrow{BC}$$
$$\overrightarrow{OG} = (1-t)\overrightarrow{OA} + (1-t)t\left(\overrightarrow{AO} + \overrightarrow{OB}\right) + t\overrightarrow{OB} + t^2\left(\overrightarrow{BO} + \overrightarrow{OC}\right)$$
$$\overrightarrow{OG} = (1-t)^2\,\overrightarrow{OA} + 2t(1-t)\overrightarrow{OB} + t^2\,\overrightarrow{OC}$$