Travaux dirigés d'infographie n°4

Cours de synthèse d'image I

— IMAC première année —

Les Courbes

Lors de cette scéance, nous abordons les principales représentations des courbes en 2D. Nous verrons ainsi les courbes de Bézier et les B-Splines.

▸ **Exercice 1.** ***Courbes de Bézier et algorithme de De Casteljau***

Les courbes de Bézier permettent de tracer des courbes "douces" passant "à proximité" d'un ensemble de points de contrôles.

- *Écrire une application openGL qui affiche la courbe de Bézier définie par les points* `(-.5,-.5.),(.5,.5)`.
  Sur la courbe :
  - *les points de contrôle sont affichés,*
  - *d'une autre couleur que la courbe,*
  - *avec la primitve* `GL_POINTS`,
  - *en augmentant la taille du point avec* `glPointSize(2.)`

  Quel est le degré de cette courbe de Bézier élémentaire ?
- *Afficher la courbe avec un dégradé du rouge clair au rouge foncé.*
- *Écrire maintenant une application qui affiche :*
  - *une ligne brisée passant par les points* `(-.5,0.),(-.25,.3),(.25,.3)`
  - *et la courbe de Bézier définie par ces points de contrôle.*
- *Écrire maintenant une application qui affiche la courbe de Bézier et le polygone de contrôle définis par un ensemble, de cardinal quelconque, de points cliqués par l'utilisateur.*

  Pour cela, on écrira une fonction récursive qui implante l'algorithme de De Casteljau.
- *Quel autre méthode avez vous vue en cours pour le calcul des points sur une courbe de Bézier ?*
- *Quel est l'inconvénient principal des courbes de Bézier pour le dessin interactif ?*

▸ **Exercice 2.** ***Courbes B-Splines***

Nous allons réaliser une application de dessin intéractive de courbes B-Splines. Les paramètres de nos fonctions B-Spline mathématiques seront :

- *n points de contrôles P_i pour i variant de 1 à n,*
- *un degré d,*
- *un ensemble de "noeuds" de taille $m = n + d + 1$ formant une partition de l'interval $[0; 1]$ (des réels $k_1 \dots k_m$ tels que $0 = k_1 \leq k_1 \leq k_2 \leq \dots \leq k_{m-1} \leq k_m = 1$).*

Pour calculer les coordonées d'un point sur la courbe, on utilisera la définition des B-Splines :

$$\overrightarrow{s(t)} = \sum_{i=1}^{n} N_{i,d}(t)\overrightarrow{P_i}$$

Où :

- *t est un paramètre avec $t \in [0; 1]$,*
- *$\overrightarrow{s(t)}$ est un vecteur pointant vers le point sur la B-Spline d'abscisse curviligne t,*
- *n est le nombre de points de contrôle,*
- *$\overrightarrow{P_i}$ est un vecteur définissant le i^{eme} point de contrôle,*
- *d est le degré de la B-Spline,*
- *$N_{i,d}(t)$ est la fonction de base d'indice i et de degré d.*

Les fonctions de base $N_{i,p}(t)$ sont définies récursivement de la façon suivante :

$$N_{i,0}(t) = \begin{cases} 1 & \text{si } k_i \leq t < k_{i+1}, \\ 0 & \text{sinon.} \end{cases}$$

$$N_{i,d}(t) = \frac{t - k_i}{k_{i+d} - k_i} N_{i,d-1}(t) + \frac{k_{i+d+1} - t}{k_{i+d+1} - k_{i+1}} N_{i+1,d-1}(t)$$

Où k_i est le i^{eme} noeud dans l'ensemble des noeuds (on dit "vecteur de noeuds").

1. *Rappeler les propriétés des fonctions Spline et des fonctions B-Spline. En particulier, précisez la particularité des fonctions de base des B-Spline les $N_{i,p}(t)$. Commentez les bornes de la somme $\overrightarrow{s(t)} = \sum_{i=1}^{n} N_{i,d}(t)\overrightarrow{P_i}$.*
2. *Écrire la fonction C*
   `float fonction_base_bspline(float *knots, int indice, int degree, float t)`
   qui évalue la valeur de la fonction $N_{i,d}$ en t.

3. *En prenant comme vecteur de noeuds* $k = \{0, 0.25, 0.5, 0.75, 1\}$ *(*$m = 4$*) tracez sur* $[0; 1]$ *les fonctions :*

   - $N_{0,0}(t)$
   - $N_{1,0}(t)$
   - $N_{2,0}(t)$
   - $N_{0,1}(t)$
   - $N_{1,1}(t)$
   - $N_{0,2}(t)$
   - $N_{1,2}(t)$

4. *Écrire la fonction C*
   `Point2D bspline(Point2D *cp, int nb_points, int degree, float *knots, float t)`

5. *Utiliser la fonction précédente pour réaliser une application de dessin de B-Spline cubique intéractive.*