ENSIMAG 3ème année / M2 MIA - GICAO
Modélisation Surfacique **6 octobre 2015**

TP Surfaces de Bézier : Interrogation de surfaces

durée 3-5h

Le but du TP est d'implémenter 2 méthodes d'interrogation de surfaces et de les tester sur des surfaces de Bézier.

Un programme Matlab est à votre disposition pour ce TP. C'est un programme graphique qui, à partir d'un ensemble de $k \cdot 16$ points de contrôle $\boldsymbol{b}_{ij}^k \in \mathbb{R}^3$, $i, j = 0, \ldots, 3$ calcule et dessine k surfaces de Bézier bicubiques

$$X^k(u,v) = \sum_{i=0}^{3} \sum_{j=0}^{3} \boldsymbol{b}_{ij}^k B_i^3(u) B_j^3(v) \ , \quad (u,v) \in [a,b] \times [c,d].$$

Vous pouvez faire ce TP en Matlab, Scilab, ou programmer en C/C++ avec OpenGL, etc. Si vous souhaitez utiliser Matlab, vous pouvez télécharger sur Chamilo l'archive *BEZIER-SURF.zip*, démarrer *matlab* et exécuter le programme principal *main.m* avec la commande *main*. Ensuite vous utiliserez ce programme comme cadre pour votre TP.

1. Génération de surfaces

La fonction *load* permet de lire des données à partir d'un fichier ascii. Nous utilisons cette fonction pour lire des points de contrôle de la surface à partir d'un fichier. Dans le répertoire se trouve un tel fichier, nommé *surface1*. Il contient les 16 points de contrôle d'un patch de Bézier.
Générez vous-même au moins 3 autres exemples de surfaces (au moins une avec $k > 1$) et sauvez les points de contrôle respectives dans 3 fichiers *surfacei*. $i = 2, 3, 4$.
Note: Vous pouvez écrire des petites routines Matlab pour générer les points de contrôle (p.ex. en échantillonnant des surfaces paramétriques connues, ou avec une méthode procédurale) et utiliser la commande *save* pour sauver les points dans le fichier ascii.

2. Champs des normales

Ecrire une fonction *bezierPatchNormal* qui pour chaque patch k calcule le champs des normales *N(:,:,:,k) = bezierPatchNormal(P(:,:,:,k),u,v)*,
où u,v sont des vecteurs Matlab de num_n valeurs de paramètre. Elle prend en entrée les 16 points de contôle et l'ensemble des valeurs de parametre en lesquels les normales N (vecteurs unitaires) sont calculées. N(:,:,:,k) est une matrice de taille num_n x num_n x 3.
La normale de la surface X au paramètre $(\bar{u}, \bar{v})$ est défini par $N(\bar{u}, \bar{v}) = \frac{X_u(\bar{u}\bar{v}) \times X_v(\bar{u},\bar{v})}{\|X_u(\bar{u},\bar{v}) \times X_v(\bar{u},\bar{v})\|}$
(voir en bas du document pour les formules des dérivées partielles X_u, X_v.)

3. Interrogation de surfaces

Ces méthodes servent à examiner visuellement la qualité d'une surface graçe à des courbes calculées sur la surface ou graçe aux visualisations des courbures (color plots).

3.1 Lignes isophotes

Les isophotes sont des lignes d'intensité lumineuse égales sur une surface pour une direction de lumière donnée. Soit $L \in I\!R^3$ la direction des rayons lumineux parallèls. Une isophote est alors définie par l'ensemble des points $X(\bar{u}, \bar{v})$ dont les paramètres $(\bar{u}, \bar{v})$ vérifient

$$< N(\bar{u}, \bar{v}), L >= c \tag{1}$$

où c est une valeur constante.

Remarque: *Comment tracer les isophotes? La façon la plus simple est d'évaluer la fonction* $I(u, v) =< N(\bar{u}, \bar{v}), L >$ *pour un ensemble assez dense de valeurs* $(\bar{u}, \bar{v})$ *et de faire un plot3 des points* $X^k(\bar{u}, \bar{v})$ *pour lesquels* $|I(\bar{u}, \bar{v}) - c| < \varepsilon$.
Une autre et meilleure possibilité est de tracer les isophotes sous forme de courbes en utilsant l'algorithm de "marching cubes" en 2D.

Travail demandé: Pour chaqu'une des 4 surfaces, *surface1,...,surface4*, calculez une dizaine d'isophotes en choisissant une direction L adaptée et en faisant varier la valeur c. Visualisez les isophotes sur les surfaces et sauvez le résultat dans un fichier image. C'est un plus si vous utilisez plusieurs méthodes de visualisation.

En option: Pertubez légèrement les points de contrôle des 4 surfaces et sauvez les points de contrôle dans des fichiers, nommés *psurface1,...,psurface4*. Calculez comme ci-dessus les isophotes avec les mêmes paramètres (view, L, c, ε) que pour les surfaces originales, puis sauvez les visualisations dans des fichiers image. Qu'observez-vous?

3.2 Curvature plot

Un plot de courbure associe à chaque point de la surface une couleur $[R, G, B] \in [0, 1]^3$ en fonction d'une courbure de la surface. Généralement on utilise la courbure de Gauss $K(u, v) = \kappa_1(u.v) \cdot \kappa_2(u, v)$ ou la courbure absolue $A(u, v) = |\kappa_1(u, v)| + |\kappa_2(u, v)|$, où κ_1, κ_2 sont les courbure principales de la surface X.

Travail demandé: Pour chaqu'une des 4 surfaces, *surface1,...,surface4*, calculez les courbure principales en unegrille dense de points sur la surface. Visualisez le curvature plot en choisissant une echelle de couleur adaptée que vous affichez sur le plot en même temps que la surface colorée.

En option: Utilisez les mêmes surfaces perturbées que dans la question précédente. Qu'observez-vous?

Travail á rendre
Le travail est à effectuer par chaque élève individuellement.
Aux dates fixées:

23/10 pour l'exercice 3.1

13/11 pour l'exercice 3.2

chaque élève dépose sur Teide une archive à son nom $nom.zip$ contenant

- les sources (matlab ou autre),
- les 4 fichiers ascii $surface1,..,surface4$ ainsi que
- les snapshots des résultats, idéalement sous forme d'un seul document pdf, sinon sous forme d'images $png/jpeg$.

Doc Matlab
La doc de Matlab est online. Il suffit d'écrire $help\ nom\_de\_commande$ pour avoir la doc sur la commande. p.ex. $help\ load$. En cliquant ensuite sur le lien "reference page for help" vous êtes redirigé sur le site web du support Matlab.

Rappel: **dérivées partielles premières** d'un patch de Bézier de degré (n, m):

$$X_u(u,v) := \frac{\partial X}{\partial u}(u,v) = \frac{n}{(b-a)} \sum_{i=0}^{n-1}\sum_{j=0}^{m}(b_{i+1,j} - b_{ij})B_i^{n-1}(u)B_j^m(v)$$

$$X_v(u,v) := \frac{\partial X}{\partial v}(u,v) = \frac{m}{(d-c)} \sum_{i=0}^{n}\sum_{j=0}^{m-1}(b_{i,j+1} - b_{ij})B_i^{n}(u)B_j^{m-1}(v),$$

où $u \in [a,b]$ et $v \in [c,d]$

Rappel: **dérivées partielles secondes** d'un patch de Bézier de degré (n, m):

$$X_{uu}(u,v) := \frac{\partial^2 X}{\partial u^2}(u,v) = \frac{n(n-1)}{(b-a)^2} \sum_{i=0}^{n-2}\sum_{j=0}^{m}(b_{i+2,j} - 2b_{i+1,j} + b_{ij})B_i^{n-2}(u)B_j^m(v)$$

$$X_{vv}(u,v) := \frac{\partial^2 X}{\partial v^2}(u,v) = \frac{m(m-1)}{(d-c)^2} \sum_{i=0}^{n}\sum_{j=0}^{m-2}(b_{i,j+2} - 2b_{i,j+1} + b_{ij})B_i^{n}(u)B_j^{m-2}(v),$$

$$X_{uv}(u,v) := \frac{\partial^2 X}{\partial uv}(u,v) = \frac{nm}{(b-a)(d-c)} \sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\Delta^{1,1}b_{ij}B_i^{n-1}(u)B_j^{m-1}(v),$$

où $u \in [a,b]$ et $v \in [c,d]$ et $\Delta^{1,1}b_{ij} = (b_{i+1,j+1} - b_{i,j+1} - b_{i+1,j} + b_{ij})$.

Rappel: Les **courbures principales** peuvent être calculées comme les valeurs propres de la matrice $H \cdot G^{-1}$, où G est la matrice de la première forme fondamental, G^{-1} son inverse, et H la matrice de la deuxième forme fondamentale. Les élémets des matrices G et H sont des simples produits scalaires des dérivée premières et secondes, ainsi que la normale.