

Modélisation de linéaires par des NURBS pour un simulateur de vols

Jean-Marc Laferté, Guillaume Daussin, Jihed Flifla

École Louis de Broglie
Département informatique
Campus de Ker Lann, 35170 Bruz, France
laferte|g.daussin|flifla@ecole-debroglie.fr

Résumé Nous présentons une application des NURBS pour la modélisation de linéaires (routes, rivières, voies ferrées etc.) pour un simulateur de vols. Le terrain est lui-même modélisée par une surface NURBS. Les données d'entrée sont des points de passage 2D de ces linéaires. On en déduit des courbes NURBS 2D, puis 3D en tenant compte de l'altimétrie. Finalement les linéaires sont modélisés par des surfaces NURBS posées sur une altimétrie définie par une autre surface NURBS. Certaines contraintes et problèmes apparaissent, notamment la gestion des intersections ou l'horizontalité des routes qui s'oppose à la pente du terrain. Nous proposons également des solutions pour les gérer. Nous présentons enfin des résultats avec plusieurs types de linéaires.

Mots clés courbes et surfaces NURBS, synthèse d'images, simulateur de vols, linéaires

1 Introduction

Les simulateurs de vols traitent une grande quantité et variété d'informations parmi lesquelles le paysage survolé prend une grande place. Le paysage, au sens où nous l'entendons, comprend le terrain que nous appellerons *altimétrie*, les arbres, les maisons, les voies routières (rues, routes départementales, autoroutes, chemins forestiers etc.), les lignes de chemins de fer, les rivières, la mer, les champs, les lignes électriques etc. Des travaux en cours dans notre laboratoire visent à modéliser toutes ces données en les classifiant par couches selon différents critères : sémantique, topologique, techniques de rendu etc. [1]. Nous nous intéressons ici à une couche particulière que nous appelons *linéaires* car désignant des « objets guidés par une courbe », telles que les rivières, routes, lignes de chemin de fer, lignes électriques entre autres. D'autres couches ont déjà été traitées, en particuliers des « objets ponctuels » tels que des arbres [2].

Aujourd'hui l'approche polygonale reste privilégiée pour modéliser le terrain au sens large, particulièrement pour les jeux vidéo. L'univers est découpé en triangles sur lesquels sont plaqués des textures spécifiques aux objets auxquels ils appartiennent. La manière dont le maillage est effectué peut différer (voir par exemple [3]) mais l'élément de base reste le triangle ou le quadrilatère. L'approche polygonale a comme principal inconvénient de générer un nombre très important de polygones afin de rendre le plus naturels possibles les méandres d'une rivière, ou les virages en épingle à cheveu d'une route de campagne. Ceci peut devenir critique pour les simulateurs de vol car, pour donner un ordre de grandeur, à une altitude de 15 kms, un pilote peut

avoir, dans des conditions météo idéales, une visibilité de plus de 400 kms ! Le nombre d'objets, donc de polygones, d'une scène aussi vaste se compte en millions ... Par ailleurs les approches polygonales ne permettent pas de traiter facilement les déformations géométriques d'une scène en temps réel contrairement aux méthodes polynômiales.

Les travaux portant sur la modélisation de linéaires sont rares, à l'exception notable des rivières. Celles-ci ont focalisé l'attention sur la problématique de l'écoulement qui induit des méthodes proches de la mécanique des fluides [4; 5] pour atteindre un réalisme accru. Cependant, ces approches coûteuses, mais justifiées pour une visualisation rapprochée des cours d'eau, ne le sont plus autant pour des simulateurs de vols dans lequel le paysage est le plus souvent observé à plusieurs kilomètres de distance. Cela est valable pour tous les linéaires.

Nos contraintes de conception sont principalement liées au respect de la topologie des modèles numériques de terrain (MNT). Les données d'entrée de nos linéaires sont leurs points de passage défini dans le système d'information géographique (SIG) international WGS84. Une première tâche est d'interpoler ces points. Nous avons optés pour un modèle NURBS Ce modèle a l'avantage de s'accorder à tous types de linéaires, d'être déformable localement et rapidement. Or dans la simulation de combats air/sol le terrain peut subir des déformations non négligeables et quasi instantanées nécessitant une modification du paysage en temps réel. Par ailleurs, aussi bonne soit l'interpolation, des ajustements locaux peuvent être nécessaires (voir section 3) tant sur l'altimétrie que sur les linéaires.

La contrainte de temps réel est importante et sévère d'une part parce que le volume d'informations à gérer est considérable. En effet comme nous l'avons déjà dit, l'univers d'un pilote volant à 40000 pieds peut couvrir plusieurs centaines de kilomètres carrés ! Dans une telle zone on peut trouver des millions d'arbres, des milliers de bâtiments, des centaines de linéaires etc. D'autre part le temps réel pour les simulateurs de vols avoisine les 60 images par seconde ... C'est cette contrainte qui nous avait poussé à paralléliser notre algorithme de simulation de forêts en l'implémentant sur GPU [2]. C'est aussi cette contrainte, parmi d'autres mentionnées plus haut, qui nous motive pour utiliser des modèles paramétriques.

2 Rappels sur les NURBS

Nous rappelons ici très brièvement quelques propriétés des NURBS qui nous ont poussés à préférer ces modèles plutôt que d'autres. Pour une description détaillée des NURBS, nous renvoyons le lecteur à un ouvrage de référence [6].

Nous nous intéressons ici uniquement à l'interpolation de points, puisqu'en effet, que ce soit pour le terrain ou pour les linéaires, nous avons en entrée de notre application une liste de points de passage issus de SIG (Systèmes d'Information Géographiques). Ces points de passage sont donc incontournables, si l'on veut rester fidèle aux données géographiques, ce qui est crucial dans certaines applications (pensez aux missions de reconnaissance ou de combat pour des simulateurs de vols militaires).

Les NURBS, pour *Non Uniform Rational B-splines*, appartiennent à la famille des courbes (ou surfaces) définies par des polynômes. L'idée sous jacente est d'imposer des contraintes à une courbe (ou surface) par le biais de points de contrôle.

On peut définir une NURBS C dans \mathbb{R}^3 d'ordre p de la manière suivante :

Soient $(P_k)_{[0,n]}$ les points de contrôles de la courbe C pondérés par les quantités $(w_k)_{[0,n]}$ et $T = (t_0, \dots, t_{n+p+1})$ un vecteur-nœuds, dont la répartition des t_i est non uniforme sur $[t_0, t_{n+p+1}]$

$$\forall t \in [0, 1], \quad \forall M \in C, \quad \overrightarrow{OM}(t) = \frac{1}{\sum_{k=0}^n w_k N_p^k(t)} \sum_{k=0}^n w_k N_p^k(t) \overrightarrow{OP_k}$$

ou

$$\overrightarrow{OM}(t) = \sum_{k=0}^n F_p^k(t) \cdot \overrightarrow{OP_k}$$

avec

$$F_p^k(t) = \frac{w_k N_p^k(t)}{\sum_{i=0}^n w_i N_p^i(t)}$$

Les N_p^k sont les polynômes de Boor se définissant de manière récursive comme suit :

$$N_p^k(t) = \begin{cases} \frac{t - t_i}{t_{i+m} - t_i} N_{p-1}^k(t) + \frac{t_{i+m+1} - t}{t_{i+m+1} - t_{i+1}} N_{p-1}^{k+1}, & p \neq 0 \\ \begin{cases} 1, & t_k \leq t < t_{k+1} \\ 0, & \text{sinon} \end{cases}, & p = 0 \end{cases}$$

Cette dernière formulation permet de ramener les NURBS à la définition générale des splines où les (F_p^k) sont les fonctions d'influence des NURBS.

Il y a au moins 2 intérêts à passer par un réseau de nœuds non uniforme :

- sur certaines zones de la surface ou courbe modélisée (par exemple le terrain), on peut souhaiter insérer de nouveaux points de contrôle pour avoir plus de détails, ou suite à un événement qui modifie la topologie du terrain par exemple. Or le nombre de points de contrôle étant directement relié au nombre de nœuds, la répartition de ces derniers ne sera pas forcément uniforme.
- une courbe avec un point d'inflexion ne peut être modélisée que par une spline non uniforme.

Le fait d'utiliser des fonctions rationnelles permet de modéliser également les coniques utiles par exemple pour simuler des ronds-points.

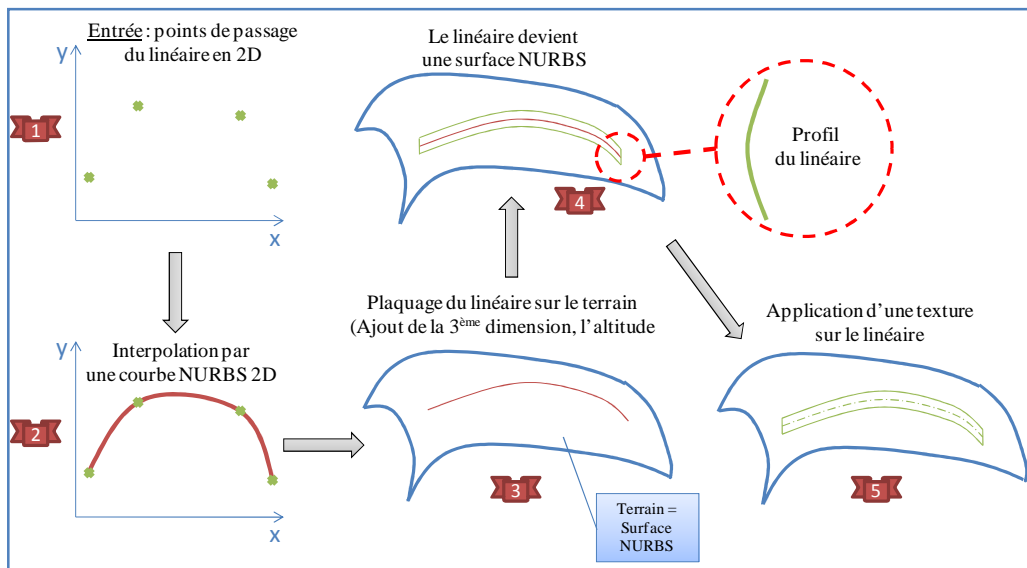


Figure 1 : synoptique de l'algorithme global

L'extension aux surfaces requiert évidemment d'utiliser un second paramètre et d'utiliser le produit tensoriel, mais la définition reste similaire et les propriétés identiques.

On peut citer parmi celles-ci :

- ✓ Interpolation des points de contrôle extrêmes
- ✓ Invariance affine : une transformation affine est effectuée en l'appliquant aux seuls points de contrôle ! Exemples : rotation, homothéties, translations.
- ✓ La surface est contenue dans l'enveloppe convexe de ses points de contrôle. Cela permet de tester itérativement si un point appartient à une surface ou de calculer le point d'intersection d'une droite avec une telle surface (notamment pour le lancer de rayons) ...
- ✓ Modification locale : on peut modifier localement la surface en déplaçant des points de contrôle, sans pour autant modifier l'ensemble de la surface (ce qui est le cas avec des surfaces de Bézier). Cette propriété nous sera très utile dans le contexte des simulateurs de vols (voir plus loin).
- ✓ Propriétés de différentiabilité dont l'ordre dépend essentiellement du degré des polynômes.

À noter également qu'il existe des algorithmes¹ très efficaces [6] permettant :

- ✓ de projeter un point sur une telle surface,
- ✓ calculer la distance de ce point à cette surface
- ✓ à partir d'un point P appartenant à la surface, on peut retrouver les paramètres de la NURBS correspondant à P
- ✓ ajouter ou enlever des points de contrôle etc.

Par rapport à notre problématique, les surfaces de type NURBS présentent 2 avantages principaux par rapport aux surfaces polygonales :

- ✓ d'une part elles nécessitent beaucoup moins de données à manipuler. En effet pour obtenir un terrain ou un linéaire réaliste et fidèle à la réalité, il faut vite manipuler un nombre très important de polygones qui surchargent la base de données et ne sont pas toujours compatibles avec les contraintes de temps réel.
- ✓ D'autre part, elles intègrent naturellement et rapidement la modification de la surface (suite à une explosion par exemple). Les surfaces polygonales requièrent un grand nombre de calculs pour recomposer la scène dans ce cas.

3 Notre approche

3.1 Généralités

L'altimétrie, c'est-à-dire le terrain, est modélisé par une surface NURBS 3D, résultat de l'interpolation de points de passage (l'algorithme d'interpolation utilisé est décrit dans [6]).

L'algorithme global est schématisé sur la Figure 1, avec les étapes de 1 à 5.

Pour chaque linéaire, nous disposons comme données d'entrée d'une liste de points de passage en 2 dimensions, que nous interpolons tout d'abord par une courbe NURBS bidimensionnelle (étapes 1 & 2). Nous appelons *génératrice* la courbe obtenue et la notons G2D. Cette courbe est alors

¹ Nous utilisons la bibliothèque SISL [7] qui implémente un certain nombre de ces algorithmes.

« posée » sur la surface NURBS tridimensionnelle (étape 3) représentant l'altimétrie en projetant les points de contrôle de G2D sur le terrain, et en ajoutant des points de contrôle si nécessaire (plus précisément si, entre 2 points de définition, la différence de courbure entre l'altimétrie (surface NURBS) et la génératrice 3D est supérieure à un certain seuil, voir [7]). À la fin de ce processus on obtient donc une courbe NURBS tridimensionnelle représentant notre génératrice (en noir sur la figure 1). Cette génératrice représente un squelette du linéaire. À partir de ce squelette sont définies les contours du linéaire, modélisés par 4 courbes NURBS, comme illustré Figure 2. À partir de ces quatre courbes, une surface NURBS est alors définie et modélise finalement le linéaire (étape 4). On utilise un « profil » qui est développé le long de la génératrice. Enfin (étape 5) on applique une texture adéquate sur le linéaire.

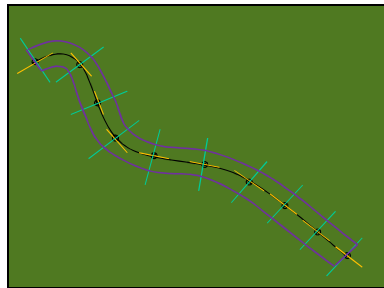


Figure 2 : modélisation du linéaire de la courbe NURBS 3D à la surface NURBS 3D
(On échantillonne la génératrice et on calcule aux points échantillonnés, les normales pour déterminer les contours.)

3.2 Intersections

La gestion des intersections de linéaires pose 2 problèmes : leur identification et leur rendu. L'identification est traitée par un algorithme efficace de calcul d'intersections de courbes NURBS [6]. En l'occurrence ce calcul est effectué au niveau des génératrices 2D pour faciliter les calculs. La troisième coordonnée (l'altitude) est ensuite ajoutée en projetant le point d'intersection sur l'altimétrie. Le résultat est satisfaisant, comme le montre la Figure 3. La nature des intersections est symbolisée par une couleur de drapeau différente. Le rendu des intersections devra tenir compte de la diversité des configurations possibles : intersections ponts/rivières, routes/voies ferrées etc. ainsi que de l'angle d'un linéaire par rapport à l'autre ...

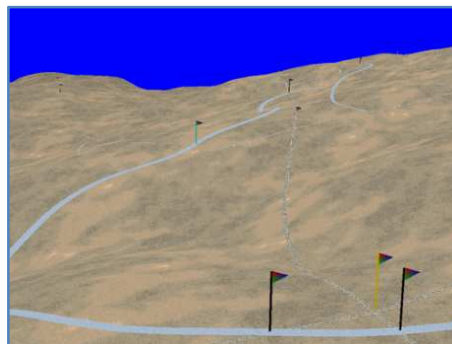


Figure 3: les intersections sont signalées par des drapeaux

3.3 Collisions

Des « collisions » entre un linéaire et le terrain peuvent se produire dans certains cas. Sur la Figure 4 gauche, on voit la route passer sous le terrain. Ceci est dû à la contrainte d'horizontalité de la route qui s'oppose à la pente naturelle du terrain. Il est donc important d'aplanir le terrain le long de la route. Il s'agit alors de déterminer les points de contrôle de la surface NURBS définissant le terrain qui sont les plus proches du linéaire considéré. Ces points sont alors projetés sur un plan qui définira le plateau sur lequel reposera le linéaire. Cette opération, appelée *flattening* (et décrite précisément dans [6], section 11.4.3) requiert relativement peu de calculs pour être mise en œuvre, comparativement à une approche polygonale qui nécessiterait de *retesseler* le terrain sur toute la surface proche du linéaire. Les figures Figure 4 et Figure 5 illustrent ce principe pour une autoroute et une rivière (l'aplanissement a été volontairement grossi par souci de visibilité).



Figure 4 : aplanissement d'un terrain sous une route

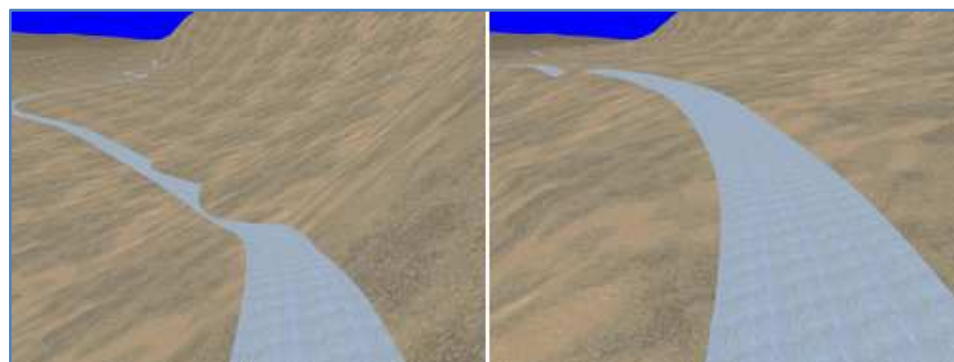


Figure 5 : déformation du terrain sous une rivière

4 Premiers résultats

Les images suivantes (Figure 6) sont des détails de la simulation d'un terrain de 40 km² (image globale de résolution 640x820) avec :

- 201x201 points de contrôle pour le terrain
- À gauche, une autoroute de largeur 18 mètres avec 50 points de contrôle
- À droite : la même autoroute plus une rivière de largeur 18 mètres avec 64 points de contrôle.

Les linéaires sont représentés par des NURBS d'ordre 5 (le degré des polynômes est donc 4), ce qui est suffisant au niveau du réalisme, étant donnée la densité du maillage formé par les points de passage.

La route épouse bien le terrain : la différence d'altitude entre le linéaire et le terrain est en moyenne de 0,2 % et au maximum de 0,4%.

Le temps de génération du terrain et des linéaires (sans ponctuels tels que des arbres) est actuellement de 8 secondes sur un processeur AMD Phenom 8600D, avec une fréquence de 2,29 GHz et 2 GO de RAM. Il faut garder à l'esprit que ces calculs ne sont effectués qu'une fois (modulo une modification du terrain due à une explosion par exemple) au cours de la simulation d'une même scène. Le rendu et l'animation sont par ailleurs gérés par OpenGL. Ces temps de génération sont optimisables, en portant par exemple une partie des calculs sur GPU.

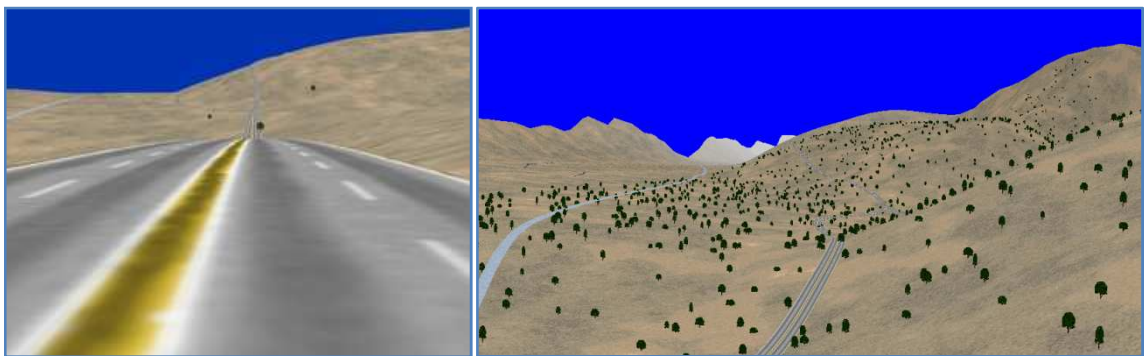


Figure 6 : simulation d'une autoroute (à gauche), autoroute et rivière (à droite)

Comme expliqué précédemment, l'un des avantages des NURBS par rapport aux modèles polygonaux notamment est de pouvoir modifier rapidement une surface. Nous l'avons illustré sur la figure suivante (Figure 7).



Figure 7 : déformation du terrain par déplacement de points de contrôle

Notons que le réalisme du terrain repose pour beaucoup sur le choix des textures, ce qui n'était pas notre préoccupation première.

5 Conclusion et perspectives

Nous avons mis en exergue la richesse des NURBS pour modéliser à la fois le terrain et différents types de linéaires, à un coût moindre (en volume de données) que celui induit par les approches

polygonaux. Notre approche permet de bien séparer le terrain et les linéaires. Ainsi, si une nouvelle route apparaît dans la base de données, il n'est pas nécessaire de recalculer toute la scène. Nous avons également montré que ces modèles permettaient de prendre en compte naturellement des déformations ponctuelles de la scène, telles que des explosions.

Il reste à implémenter les solutions permettant d'éviter les collisions entre linéaires ou entre un linéaire et le terrain. Il reste également à gérer le rendu des croisements de linéaires (carrefours, ronds-points, ponts etc.), les ronds-points pouvant se représenter par les NURBS en tant que coniques ... Enfin, les calculs peuvent être accélérés de manière importante en implémentant tout ou partie de l'algorithme sur des processeurs graphiques, à condition de paralléliser l'algorithme.

Références

- [1]. Daussin, G. et al. A Layered Multi Modelization Multi Visualization Concept for Virtual Universes Applied to the Flight Simulator Context. *Rapport interne*. 2009. Consultable sur demande à laferte@ecole-debroglie.fr.
- [2]. Laferté, JM et al. Real-time Forest Simulation for a Flight Simulator using a GPU. *International Conference on Information & Communication Technologies: From Theory to Application*. Avril 2008. 2d Best Paper Award..
- [3]. Loviscach, J. Paving Procedural Raods with Pixel Shaders. WSCG. 2005, pp. 39-46.
- [4]. Yu, Q., et al. Scalable Real-Time Animation of Rivers. *Computer Graphics Forum (Proceedings of Eurographics)*. Mars 2009, Vol. 28, 2.
- [5]. Blanc, Carole. Techniques de modélisation et de déformation de surfaces pour la synthèse d'images. Bordeaux 1 : s.n., 1994. Thèse de doctorat.
- [6]. Piegl, L. et Tiller, W. The NURBS Book. 2ème édition. s.l. : Springer, 1997.
- [7]. Dokken, T. et al. Near best approximation of circles by curvature-continuous Bézier curves. *Computer Aided Geometric Design*. 1990.
- [8]. <http://www.sintef.no/Informasjons--og-kommunikasjonsteknologi-IKT/Anvendt-matematikk/Fagomrader/Geometri/Prosjekter/The-SISL-Nurbs-Library/SISL-Homepage/>. Bibliothèque SISL. [En ligne]
- [9]. Belhadj, F. et Audibert, P. Modeling Landscapes with Ridges and Rivers. *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*. Nov. 2005.
- [10]. De Casteljau, Paul. Courbes et surfaces à pôles. Citroën. 1967. Rapport technique.