109

ＮＵＲＢＳを用いた曲面モデル

北海道大学工学部　○田中　文基、岸浪　建史

１．はじめに

近年、曲面補間式ではＮＵＲＢ(Non Uniform Rational B-Spline)形式が注目されつつある。ＮＵＲＢは、有理ベジェ曲面や有理クーンズ曲面のスーパーセットであり、形状表現の多様性にその特徴がある。しかし、ＮＵＲＢ特有のデータとしてノットベクトルがあり、また、制御点の保持方法もむずかしい。そこで本研究は、ＮＵＲＢ　Ｓｕｒｆａｃｅ（ＮＵＲＢＳ）のデータ構造、ノットベクトルや制御点の保持方法について報告する。

２．ＮＵＲＢＳ [1) 2)]

工業製品の設計において、曲面補間式は形状を表現する上で重要な役割を担う。特に、形状表現の多様性は重要なファクターである。そのなかで、Ｂスプライン曲面は、最も形状表現の多様性が高い補間形式と言われている。ＮＵＲＢ補間形式は、有理Ｂスプライン補間ともいわれＢスプライン形式に有理多項式表現を導入したものである。

一般のＮＵＲＢＳの補間形式は（１）式で表される。

$$\mathbf{S}(u,v)=\sum_{i=0}^{m}\sum_{j=0}^{n} Ni,k(u)\,Nj,l(v)\,\mathbf{P}ij \qquad (1)$$

ただし、$\mathbf{S}(u,v)$は補間された曲面上の点の４次元同次座標ベクトル、$\mathbf{P}ij$はＮＵＲＢＳを定義する制御点の４次元同次座標ベクトル、$Ni,k(u)$、$Nj,l(v)$はそれぞれ階数がk,lのＢスプライン関数とする。このパッチと制御点との関係を図１に示す。

Ｂスプライン関数はノットベクトルにより定義される。例えば、図２はノットベクトルを[0,1/7,2/7,3/7,4/7,5/7,6/7,1]とした場合のＢスプライン関数で、図３はノットベクトルを[0,0,0,0,1,1,1,1]とした場合のＢスプライン関数である。さて、このノットベクトルの選択方法はさまざまな種類がある。ＮＵＲＢ補間形式では、各ノット間の間隔は一般に不均一であるとされている。本研究ではＢスプライン関数の階数を４すなわちＢスプライン関数の次数を３として考え、さらには$\mathbf{P}i0$、$\mathbf{P}in$、$\mathbf{P}0j$、$\mathbf{P}mj$（i=0,1,..,m、j=0,1,..,n）の制御点群が各境界線を定義し、$\mathbf{P}00$、$\mathbf{P}0n$、$\mathbf{P}m0$、$\mathbf{P}mn$を各パッチの頂点になるようにノットベクトルを設定することとする。その場合、ノットベクトルＴは（２）式で与えられる。

Ｔ＝[0,0,0,0,t1,t2....,ts,1,1,1,1]　（２）

ここで、ノットベクトルの要素のうち0、1間の個数をs個とすると、制御点数はs+4個必要である。なお、s=0の場合Ｂスプライン補間形式はベジェ補間形式となる。以上の関係を表１で示した。

次にＮＵＲＢ補間形式を用いた曲面（ＮＵＲＢ　Ｓｕｒｆａｃｅ）のデータ構造について考察する。

３．ＮＵＲＢＳのデータ構造

ＮＵＲＢＳモデルに対し、位相構造と幾何構造を定義する必要がある。以下では、それらについて述べる。

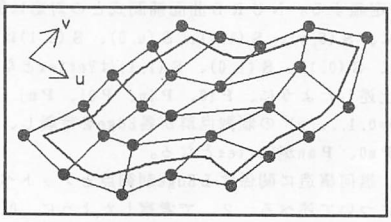

図１．ＮＵＲＢＳパッチ

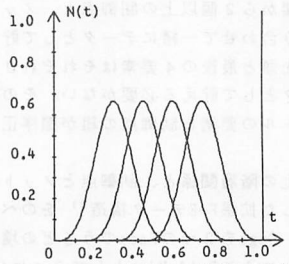

図２．Ｂスプライン関数（１）

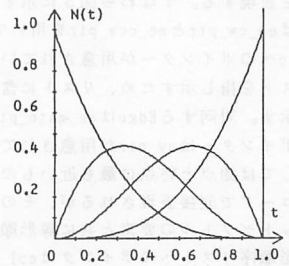

図３．Ｂスプライン関数（２）

ノットベクトル	s	制御点数
[0,0,0,0,1,1,1,1]	0	4
[0,0,0,0,t1,1,1,1,1]	1	5
[0,0,0,0,t1,t2,1,1,1,1]	2	6
[0,0,0,0,t1,t2,...,ts,1,1,1,1]	s	s＋4

表１．ノットベクトルと制御点の関係

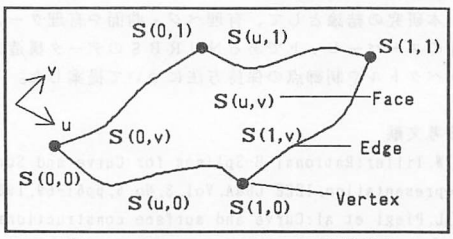

図４．ＮＵＲＢＳモデルの位相構造

ＮＵＲＢＳモデルに対して、次のような位相構造的階層関係を定義する。すなわち、各曲面パッチをFace、Faceの境界をEdge、Faceの各頂点、すなわちEdgeとEdgeの交点をVertexと定義する。最後にVertex以外の制御点をControl pointと定義する。ＮＵＲＢ曲面補間式との対応は図４に示すように、Ｓ(0,v)、Ｓ(1,v)、Ｓ(u,0)、Ｓ(u,1)はEdge、Ｓ(0,0)、Ｓ(0,1)、Ｓ(1,0)、Ｓ(1,1)はVertexとなる。また、上述したように、Ｐi0、Ｐin、Ｐ0j、Ｐmj（i=0,1,..,m、j=0,1,..,n）の制御点群が各Edgeを定義し、Ｐ00、Ｐ0n、Ｐm0、ＰmnがVertexとなる。

次に、幾何構造に関係するEdge制御点とノットベクトルの関係について述べる。２．で考察したように、0、1間のノットベクトルの要素の個数は、制御点数－４である。そのため、両端から２個以上の制御点と、ノットベクトルの要素とを組み合わせて一緒にデータとして貯える。ノットベクトルの先頭と最後の４要素はそれぞれ０、１であるために、データとして貯える必要がない。そのため、中間のノットベクトルの要素と制御点の組が順序正しく貯えていればよい。

最後に以上の階層関係と、制御点とノットベクトルとの関係を考慮した拡張F-Eデータ構造[3]をのべる。拡張F-Eデータ構造は、Edgeを２つのFaceのうちどの境界として用いられているかによりHalf-Edgeとして２つに分け、Edgeの隣接関係情報を表現する。すなわち図５に示すように、Face回りのEdgeはee_cw_ptrとee_ccw_ptrを用いて表現されており、参照Faceへのポインターが用意されている。Faceからは、そのリストを指し示すため、リストに含まれるEdgeの一つを指し示す。対向するEdgeはee_mate_ptrで表現される。Vertexへのポインターはev_ptrが用意されている。Edgeの制御点に対しては始点と終点に最も近いものについてはEdgeデータレコードで直接表現されるが、その他の制御点は、対応するノットベクトルの要素と共に線形順序リストを構成する。線形順序リストへのポインタはepl_ptrで、リスト構造のため、制御点の数は任意である。Face制御点に対しては、図６に示すようなEdgeに対応して一回り内側の制御点列とその制御点列の多層構造を考える。以上のデータ構造のＣ言語による表現を、図７に示した。

提案するデータ構造の利点として、

１）Edgeベース、Faceベースのモデリングが可能。

２）Face、Edgeの制御点を曖昧さなく取り出すことが可能。

３）ノットベクトルのデータ構造による保持が可能。

があげられる。

４．結論

本研究の結論として、有理ベジェ曲面や有理クーンズ曲面のスーパーセットであるＮＵＲＢＳのデータ構造、ノットベクトルや制御点の保持方法について提案した。

参考文献

1)W.Tiller:Rational B-Splines for Curve and Surface Representation,IEEE CG&A,Vol.3,No.6,pp61-69,1983

2)L.Piegl et al:Curve and surface constructions using rational B-splines,CAD,Vol.19,No.9,pp485-498,1987

3)田中他：拡張V-E、拡張F-Eデータ構造による曲面モデル、1989年度精密工学会秋季大会学術講演会講演論文集

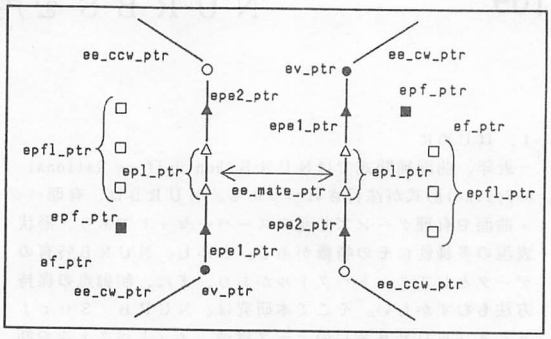

図５．拡張Ｆ－Ｅ構造

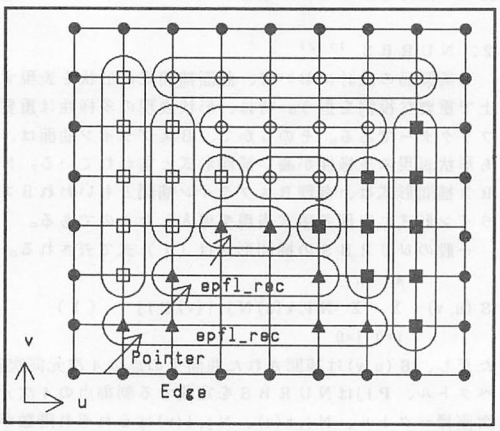

図６．EdgeとFace制御点の関係

```
Ｃ言語表現
struct face_rec{
  int feu_ptr;}      /*FaceからEdge_Useへのポインタ*/
struct eu_rec{
  int ee_cw_ptr;     /*CW方向Edge_Useへのポインタ*/
  int ee_ccw_ptr;    /*CCW方向Edge_Useへのポインタ*/
  int ee_mate_ptr;   /*Mate Edgeへのポインタ*/
  int ev_ptr;        /*Vertexへのポインタ*/
  int ef_ptr;        /*Faceへのポインタ*/
  int epe1_ptr;      /*Edge制御点(始点)へのポインタ*/
  int epe2_ptr;      /*Edge制御点(終点)へのポインタ*/
  int epl_ptr;       /*Edge制御点リストへのポインタ*/
  int epf_ptr;       /*Face制御点(始点)へのポインタ*/
  int epfl_ptr;      /*Face制御点リストへのポインタ*/
struct epl_rec{
  int next_epl_ptr; /*次のリストへのポインタ*/
  float knot;        /*ノットベクトルの要素*/
  int eplp_ptr;}     /*制御点へのポインタ*/
struct epfl_rec{
  int is_epf_rec;    /*内側制御点リストへのポインタ*/
  int epfp_ptr;      /*Face制御点へのポインタ*/
  int epfln_ptr;}    /*続く制御点リストへのポインタ*/
struct epfln_rec{
  int next_epfl_ptr;/*次のリストへのポインタ*/
  int epfp_ptr;}     /*Face制御点へのポインタ*/
struct v_rec{
  float coord[4];} /*Vertexの座標値と重み*/
struct cp_rec{
  float coord[4];} /*制御点の座標値と重み*/
```

図７．拡張Ｆ－Ｅ構造のＣ言語表現