多边形物体的精确 B-样条自由变形*

冯结青 彭群生

(浙江大学 CAD&CG 国家重点实验室 杭州 310027)

E-mail: jqfeng@cad.zju.edu.cn

摘要 在计算机动画与几何造型中,自由变形是一种重要的几何形状修改方法.该文从移位算子和函数复合的观点探讨一种方法,即当被变形物体用三角片表示、变形工具为 B-样条参数体时,变形后的物体可以精确地表示为一组三角 Bézier 曲面片,其次数为 B-样条参数体 3 个方向的次数之和.此方法的核心在于自由变形是作用在三角片上,而不是顶点上,所以解决了多边形物体 B-样条自由变形的点采样问题.

关 键 词 自由变形,函数复合,移位算子,三角 Bézier 曲面片.

中图法分类号 TP391

在计算机动画和几何造型中,几何形状的修改是一个重要的研究课题.用户可以通过交互地拉动物体的顶点或控制顶点或者对其施加变换的方法改变物体的几何形状,但这些方法所产生的效果是有限的,用它们难以得到复杂的形状.当顶点或控制顶点的数目较多时,这种直接操作方式就显得十分繁琐.空间变形方法则较好地解决了这个问题[1].

Barr 首先提出了整体与局部变形的概念[2].在他的方法中,变换矩阵不再是常数,而是空间位置的函数,因而变形可独立于物体的具体表示形式.自由变形(free-form deformation,简称 FFD)的概念是由 Sederberg 和 Parry 首先提出来的[3],这种方法提供了一个更为一般的空间变形方法框架:待变形物体首先被嵌入一个中间空间,当中间空间的形状发生变化时,变形传递给嵌入其中的物体.在文献[3]中,该空间为一个三变元张量积 Bézier 体.后来人们推广了 FFD 的概念,陆续提出了 B-spline FFD[4],Extended FFD[5],Rational FFD[6],NURBS FFD[7]以及 Continuous FFD[8]等方法.所有这些 FFD 方法都遵循相同的处理步骤:

(1) 定义一个参数体,包括参数空间和控制顶点网格;

(2) 将物体映射到参数空间中;

(3) 通过编辑控制顶点或其他方式改变参数体的形状;

(4) 嵌在参数空间中的物体作相应变形.

其他种类的 FFD 推广还包括:Animated FFD[9]、直接操作 FFD[10]、任意拓扑网格 FFD[11]、Dirichlet FFD[12]等,这些方法的处理步骤与上述 4 步略有不同.如果将 FFD 方法的空间控制网格看作二维变形工具的话,则曲面控制变形方法[13]、轴变形方法[14,15]和基于空间点及其位移约束的空间变形方法[16],分别具有二维、一维、零维的变形控制工具.在文献[17]中,Bechmann 对上述具有不同拓扑维数变形工具的变形方法进行了详细的分析和比较.

上述所有空间变形方法都是独立于物体的具体几何表示形式的,即变形作用在物体的顶点或控制顶点上,而与物体的拓扑无关.理论上,只有当变形作用在物体的每一点上时,变形才是精确的.显然,如果用目前的

---

* 本文研究得到国家自然科学基金资助.作者冯结青,1970 年生,博士,助理研究员,主要研究领域为几何造型,计算机动画,科学计算可视化.彭群生,1947 年生,博士,教授,博士生导师,主要研究领域为真实感图形,计算机动画,三维几何造型,红外图像形,工程图纸扫描识别.

本文通讯联系人:冯结青,杭州 310027,浙江大学 CAD&CG 国家重点实验室

本文 1998-09-21 收到原稿,1998-12-01 收到修改稿

方法,这种理论化的结果是不可能得到的,原因是物体在计算机中只能以离散的形式表示.当表示物体的顶点或控制顶点较稀疏时,用当前的方法就难以得到理想的变形结果,最差的情形是物体保持不变,或物体的拓扑关系发生变化.所以在实际应用中,变形物体的采样是一个重要的问题.迄今为止,只有文献[4]涉及了这个问题.在他们的方法中,系统自动地计算多边形边的中点的精确变形与线性插值之间的误差,当误差大于阈值时,当前多边形就会以插入顶点的方式加密采样,直至所有边的采样误差小于给定值.这种方法的一个不足之处在于,采样方式不能保证检测到所有奇异点,而这些奇异点往往刻画了变形的特征.

在本文中,我们借助于移位算子的概念,用函数复合的方法解决了多边形物体的 B-样条自由变形的采样问题.在我们的方法中,首先将 B-样条参数体通过节点插入算法转化为分块连续的 Beizer 参数体;然后,根据 B-样条参数体的节点向量对多边形物体进行剖分和重新三角化,使剖分和重新三角化后物体的每一个三角片严格地位于某个 Bézier 参数体之内;最后通过函数复合三角片与相应的 Bézier 参数体,得到三角片的精确变形结果,即一个次数为 B-样条体各个方向的次数之和的三角 Bézier 曲面片.由于应用了移位算子,复合过程在形式上变得清晰简明.三角 Bézier 曲面片控制顶点的计算则通过广义 de Casteljau 算法,众所周知,这是数值稳定的.所有上述过程都无需用户的干预,而是由算法自动完成的.

本文第 1 节介绍移位算子以及一种 B-样条参数体的定义方法.第 2 节介绍多边形物体的剖分和三角化等预处理过程.第 3 节介绍基于移位算子和广义 de Casteljau 算法的函数复合算法.最后是对算法的讨论以及实验结果.

1 移位算子及广义 de Casteljau 算法

结合移位算子,Bernstein 多项式可以用一种更为清晰简洁的方式表达,许多关于 Bézier 曲线、曲面的性质可以简洁地推导出来[18].记 $r(u,v,w)$ 为一个 Bézier 参数体,其定义为

$$r(u,v,w)=\sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\sum_{k=0}^{n_w} r_{ijk}B_{i,n_u}(u)B_{j,n_v}(v)B_{k,n_w}(w), \tag{1}$$

其中 $r_{ijk}\in R^3$.记 E_u, E_v, E_w, I 为移位算子,定义如下:

$$E_u r_{ijk}=r_{i+1,j,k},\quad E_v r_{ijk}=r_{i,j+1,k},\quad E_w r_{ijk}=r_{i,j,k+1},\quad I r_{ijk}=r_{ijk}.$$

应用移位算子,方程(1)可以重新表示为

$$r(u,v,w)=\left[(1-u)I+uE_u\right]^{n_u}\left[(1-v)I+vE_v\right]^{n_v}\left[(1-w)I+wE_w\right]^{n_w} r_{000}. \tag{2}$$

式(1)与式(2)的等价性可以通过二项式展开定理得到证明,在此从略.

下面我们给出在 FFD 方法中 B-样条体的一种定义方式.记被变形物体在物体坐标系中的包围盒为 $[X_{min},X_{max}]\times[Y_{min},Y_{max}]\times[Z_{min},Z_{max}]$.通过用户交互或系统自动生成的方式,根据这个包围盒生成的 B-样条体的节点向量 u, v, w 定义为

$$u=\{u_0,\ldots,u_{k_u},u_{k_u+1},\ldots,u_{n_u},\ldots,u_{n_u+k_u}\},$$

其中 k_u 为 B-样条体 u 方向的次数,n_u 为 u 方向的控制顶点的个数,端节点为 k_u 重节点.v,w 方向的节点向量 v,w 具有类似定义.由上述节点向量确定的各参数方向次数分别为 k_u,k_v,k_w 的 B-样条体 $P(u,v,w)$ 为

$$P(u,v,w)=\sum_{i=0}^{n_u}\sum_{j=0}^{n_v}\sum_{k=0}^{n_w} P_{ijk}N_{i,k_u}(u)N_{j,k_v}(v)N_{k,k_w}(w).$$

采用上述节点向量的定义方法的优点是,当我们把待变形物体映射到 B-样条体的参数空间中时,物体的坐标值与物体在参数空间中的局部坐标等价,从而节省了映射时的计算量.然后,B-样条体通过节点插入算法转化为分块连续的 Bézier 参数体[19].

2 物体的剖分和重新三角化

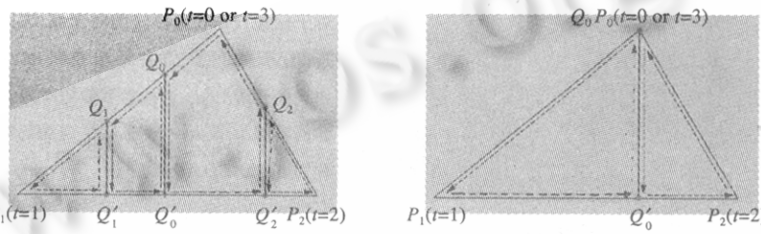
在本文中,我们假设待变形物体由三角平面片组成,这是因为三角片可以保证共面性.如果输入物体不是三角化的多面体,我们可以首先对其三角化.本节所描述的剖分与重新三角化是在 B-样条体的参数空间中进

行的,为简化起见,将向量 $\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}$ 重新记作 $\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$;将 u_i,v_j,w_k 记作 x_i,y_j,z_k.

记 $P_0P_1P_2$ 为物体上的一个三角片,如图 1(a)所示,三角形的三条边 P_0P_1,P_1P_2,P_2P_0 可以作线性参数化.这里我们不对它们作统一的[0,1]参数化,而是如下定义:

$$\begin{cases} e_0=(1-t)P_0+tP_1, & t\in[0,1];\\ e_1=(2-t)P_1+(t-1)P_2, & t\in[1,2];\\ e_2=(3-t)P_2+(t-2)P_0, & t\in[2,3]. \end{cases}$$

假定 u 方向节点所在平面 $x=x_i(i=i_0,\ldots,i_1)$与三角片 $P_0P_1P_2$ 相交,我们的目的是求出平面与三角片的交点,并在保持交线的约束下对三角片重新三角化,使结果中每一个三角片位于一个 Bézier 参数体之内.为此,我们首先计算平面与三角片的交点,所有的交点与三角片的 3 个顶点形成一个单向循环链表;对于这个链表,我们通过适当的跟踪算法得到一系列的平面多边形,简称为 Loop;最后对边数大于 3 的 Loop 进行三角化.上述过程依次对 v,w 方向类似地进行.下面我们就详细介绍上述算法.



(a) 有 3 对交点,它们用圆点表示并用线段两两相连,此图中的循环链表为 $P_0Q_0Q_1P_1Q_1'Q_0'Q_2'P_2Q_2P_0$;根据我们的 Loop 生成算法所得到的 4 个 Loop 分别为:$P_0Q_0Q_0'Q_2'Q_2$,$Q_0Q_1Q_1'Q_0'$,$Q_1P_1Q_1'$,$Q_2'P_2Q_2$.

(b) 这是一种奇异情况,三角片顶点 P_0 首先从 Vlist 中删除;最终得到的 Vlist 为 $Q_0P_1Q_0'P_2Q_0$;相应的两个 Loop 为 $P_1Q_0'Q_0$,$Q_0'P_2Q_0$.

图 1　三角片的剖分

2.1 求交及循环链表的生成

为了便于描述算法,我们首先介绍一个用于存储三角片顶点及三角片与平面交点的数据结构,如下所示.

```
Struct vertex {
    float x,y,z;              /* space position coordinates */
    float t;                  /* t parameter corresponding to edge parameterizations */
    short int Vflag;          /* "1" if triangle vertex; "2" if intersection points */
    struct vertex  *link;     /* link between two intersections, NULL if triangle vertex */
    struct vertex  *next;     /* for linked list */
}
```

在进行求交之前,P_0,P_1,P_2 首先根据其 t 值形成一个循环链表,记作 Vlist,这 3 个点的 Vflag 置为 1,link 置为 NULL,一般情况下,一个平面与一个三角片的交点数为 2,此时,这两个关联的交点通过 link 相连,其 Vflag 置为 2.然后我们根据这两个交点的 t 值大小将其插入链表 Vlist 中.这样,三角片的顶点及平面与其交点根据其 t 值大小就形成了一个有序的循环链表,如图 1(a)所示,属于同一个平面的两个交点通过 link 相连,这个信息对我们下一步生成 Loop 是很重要的.

下面我们来考虑一些奇异情形.当平面与三角片交于一点或一条边时,平面被视作与三角片无交点.如果平面交于三角片的一个顶点及该顶点所对的边,如图 1(b)所示,此时,我们首先将该三角片的顶点从 Vlist 中删除,然后再将两个交点插入 Vlist.

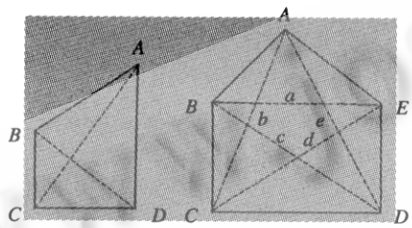
2.2 Loop的生成算法

在本节中,我们介绍如何由循环链表 Vlist 生成 Loop,如图 1 所示.一个 Loop 的生成从一个 startV 开始,startV 是链表 Vlist 中 Vflag 值为 1 的任意一个点.如果 Vlist 中的某个顶点被访问过了,则其 Vflag 值减去

1;当生成了所有的 Loop 之后,Vlist 中所有元素的 Vflag 值均为 0.这是由于所有的三角片顶点被访问一次,而交点被访问两次.访问 Vlist 中元素的顺序是这样确定的:Loop 的第 1 条边一定为三角片的边,即下一个顶点由 next 指针决定,然后,一般情形下是 next 和 link 指针交替使用.如果当前点是三角片的点,则下一个点一定是交点(由 next 指针确定),即使前一个所用的指针是 next.这个算法的特点是生成的所有 Loop 的顶点顺序与原来三角片顶点的顺序一致,也就是说如果原来的三角片的 3 个顶点是顺时针方向的,则所有 Loop 也是顺时针方向的.这一点在绘制时很重要,因为它决定了后面的三角 Bézier 曲面片与对应的三角片具有相同趋势的外法向.

2.3 重新三角化

根据第 2.2 节中的算法,共有 3 类 Loop 生成,它们的边数分别为 3,4,5.对于边数为 4 或 5 的 Loop,我们根据边长和最小的原则将其三角化.对于四边 Loop,选择较短的对角线将其三角化;对于五边 Loop,每一个顶点可以发出两条对角线,在这 5 组对角线中,我们选择具有边长和最短的顶点进行三角化,如图 2 所示.



对于四边 Loop,选择短的对角线,这里是 *BD*,所生成的三角片为 *ABD* 和 *BCD*.如果是五边 Loop,选择具有最短对角线和的顶点,如果 *b*+*e* 最短,则生成的三角片为 *ABC*,*ACD*,*ADE*.

图 2 三角化 Loop

3 平面三角片和 Bézier 参数体的函数复合

待变形物体经过剖分和重新三角化之后,物体的每一个三角片都位于某个 Bézier 参数体之中.记 $P_0P_1P_2$ 为一个三角片,它的 3 个顶点相对于 Bézier 参数体的局部坐标记为 (u_0,v_0,w_0),(u_1,v_1,w_1),(u_2,v_2,w_2).这个三角片可以通过重心坐标参数化为

$$\begin{cases} u(x,y,z)=u_0x+u_1y+u_2z, \\ v(x,y,z)=v_0x+v_1y+v_2z, \\ w(x,y,z)=w_0x+w_1y+w_2z. \end{cases} \tag{4}$$

其中(*x*,*y*,*z*)为重心坐标.将上述参数化方程式(4)代入式(2),我们就可以得到三角片 FFD 的精确结果.这里的核心问题是如何设计一个算法来快速地求取函数复合结果.

DeRose 等人曾研究过 Bernstein 多项式的函数复合算法[20,21],这里我们采用与其不同的方法来探讨这个问题.我们借助于移位算子的概念,可以使 Bernstein 多项式的函数复合问题的推导变得更加简洁和易于理解.下面我们给出 1 次三角 Bézier 曲面片(即三角平面片)与张量积 Bézier 参数体的函数复合算法.将上面的重心参数化方程组式(4)代入式(2)得

$$\begin{aligned} \boldsymbol{R}(x,y,z) &= \boldsymbol{r}(u(x,y,z),v(x,y,z),w(x,y,z)) \\ &= [(1-u(x,y,z))I+u(x,y,z)E_u]^{n_u}[(1-v(x,y,z))I+v(x,y,z)E_v]^{n_v}[(1-w(x,y,z))I+w(x,y,z)E_w]^{n_w}\boldsymbol{r}_{000} \\ &= [((1-u_0)I+u_0E_u)x+((1-u_1)I+u_1E_u)y+((1-u_2)I+u_2E_u)z]^{n_u}\bullet \\ &\quad [((1-v_0)I+v_0E_v)x+((1-v_1)I+v_1E_v)y+((1-v_2)I+v_2E_v)z]^{n_v}\bullet \\ &\quad [((1-w_0)I+w_0E_w)x+((1-w_1)I+w_1E_w)y+((1-w_2)I+w_2E_w)z]^{n_w}\boldsymbol{r}_{000}. \end{aligned}$$

记 $Au_i=(1-u_i)I+u_iE_u$,$Av_i=(1-v_i)I+v_iE_v$,$Aw_i=(1-w_i)I+w_iE_w$,i=0,1,2.将这些记号带入上面的表达式,$\boldsymbol{R}(x,y,z)$ 可以简化为

$$\begin{aligned}R(x,y,z)&=\left[Au_0x+Au_1y+Au_2z\right]^{n_u}\left[Av_0x+Av_1y+Av_2z\right]^{n_v}\left[Aw_0x+Aw_1y+Aw_2z\right]^{n_w}r_{000}\\&=\left[\sum_{i_u+j_u+k_u=n_u}Au_0^{i_u}Au_1^{j_u}Au_2^{k_u}B^{n_u}_{i_u,j_u,k_u}(x,y,z)\right]\left[\sum_{i_v+j_v+k_v=n_v}Av_0^{i_v}Av_1^{j_v}Av_2^{k_v}B^{n_v}_{i_v,j_v,k_v}(x,y,z)\right]\cdot\\&\quad\left[\sum_{i_w+j_v+k_w=n_w}Aw_0^{i_w}Aw_1^{j_w}Aw_2^{k_w}B^{n_w}_{i_w,j_w,k_w}(x,y,z)\right]r_{000}\\&=\sum_{i+j+k=N}R_{ijk}B^N_{i,j,k}(x,y,z).\end{aligned}$$

其中 $N=n_u+n_v+n_w$, R_{ijk} 为

$$\frac{\sum_{\substack{i_u+i_v+i_w=i\\j_u+j_v+j_w=j\\k_u+k_v+k_w=k}}\left(C^{n_u}_{i_u,j_u,k_u}C^{n_v}_{i_v,j_v,k_v}C^{n_w}_{i_w,j_w,k_w}Au_0^{i_u}Au_1^{j_u}Au_2^{k_u}Av_0^{i_v}Av_1^{j_v}Av_2^{k_v}Aw_0^{i_w}Aw_1^{j_w}Aw_2^{k_w}r_{ijk}\right)}{C^N_{i,j,k}}. \tag{5}$$

根据上述推导可知,复合结果是一个次数为 N 的三角 Bézier 曲面片,它的控制顶点可以通过广义 de Casteljau 算法求得.这里所得到的三角 Bézier 曲面片的法向与原来的物体的外法向趋势一致,这个性质可由我们的剖分和重新三角化算法保证.

4　算法讨论与实验结果

由于我们对物体所作的剖分和重新三角化过程是与物体的三角片的大小无关的,所以在重新三角化的结果中,会产生一些较小的三角片,对于这些较小的三角片,我们没有必要将其作精确的自由变形,因为它们不存在采样问题.所以在具体实现该算法时,可以设定一个阈值,当三角片的三条边均小于阈值时,该三角片只作普通的自由变形,而非本文提出的精确自由变形,从而可以节省计算量.但是,这可能会增加处理面片裂缝的负担.

另外一个可以节省计算量的技巧是:对于每个三角片,检测其是否位于某个参数平面中,即看其 3 个顶点的局部坐标是否有一个坐标分量相等,如果相等,我们可以简化计算控制顶点的过程.例如,如果某个三角片的 3 个顶点 u 分量均为 $u=u^*$,那么结果的三角 Bézier 曲面片的次数应为 k_v+k_w,而不是 $k_u+k_v+k_w$.

在目前的几何造型或计算机动画系统中,在用户交互过程中,为了提高显示速度,系统一般都显示其线框图.对于本文提出的算法,我们也可以计算物体的精确的线框变形,即计算线段与 Bézier 参数体的复合,其结果为一条 Bézier 曲线.限于篇幅,我们在此省略该推导.

我们已经在 SGI Indy 图形工作站上实现本算法.在图 3~5 中,(a)为初始物体;(b)为直接的自由变形结果,即变形作用在顶点,而非三角片上;B-样条参数体的控制顶点及网格在图中用点和虚线表示;(c)为物体在参数空间经过剖分和重新三角化的结果,其中虚线表示 B-样条体节点向量;(d)为精确变形结果.

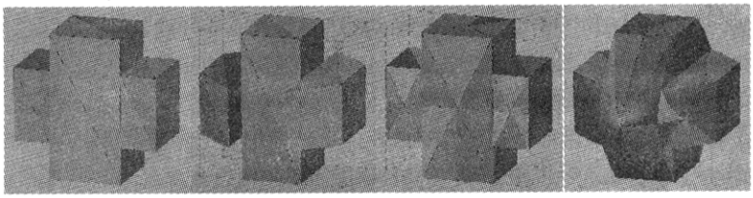
5　结　论

在本文中,我们借助于移位算子推导出了多边形物体的精确 B-样条自由变形结果.首先我们提出了一种适合于本文问题的物体的剖分和重新三角化方法,然后研究了线形 Bézier 三角曲面片与张量积 Bézier 体的函数复合,即自由变形作用于三角片,并给出了具体算法.实验结果验证了算法的正确性.本文提出的算法可以容易地集成入已有的动画或几何造型系统,从而丰富了几何造型手段.我们认为进一步的研究方向包括:

(1) 将本算法可以直接推广到 RFFD[6],NURBS FFD[7],CFFD[8];

(2) 曲面表示物体的 FFD 精确变形;

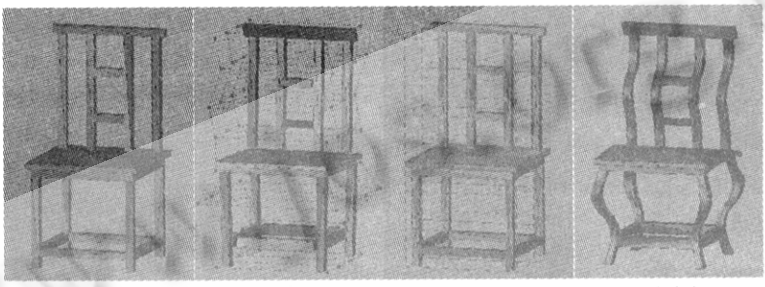
(3) 其他类型的精确变形结果,主要是基于点约束的空间变形、轴变形以及曲面控制自由变形方法.



(a) 初始物体 (b) 直接变形 (c) 物体剖分和三角 (d) 精确变形

B-样条体的次数为 2×1×2,控制顶点为 4×2×4.由于采样不足,在(b)中,初始时共面的几个三角片变形后不再共面,甚至不连续.而在(d)中,变形物体是光滑的.

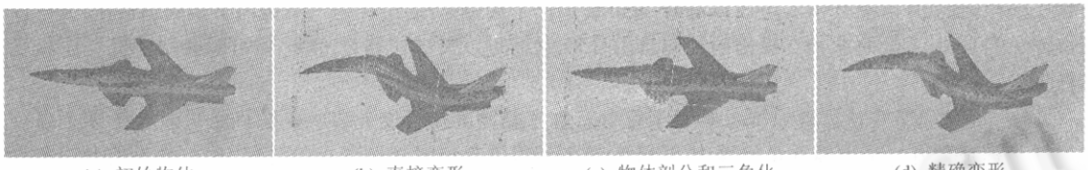
图 3



(a) 初始物体 (b) 直接变形 (c) 物体剖分和三角 (d) 精确变形

B-样条体的次数为 1×2×2,控制顶点为 2×8×4.

图 4



(a) 初始物体 (b) 直接变形 (c) 物体剖分和三角化 (d) 精确变形

B-样条体的次数为 2×1×1,控制顶点为 4×2×2.

图 5

参考文献

1 Bechmann D. Space deformation survey. Computers and Graphics, 1994,18(4):571~586

2 Barr A. Global and local deformation of solid primitives. ACM Computer Graphics (Siggraph'84), 1984,18(3):21~30

3 Sederberg T, Parry S. Free-form deformation of solid geometric models. ACM Computer Graphics (Siggraph'86), 1985,20(4): 537~541

4 Griessmair J, Purgathofer W. Deformation of solids with trivariate B-splines. In: Hansmann *et al* eds. Proceedings of Eurographics'89. North-Holland: Elsevier Science Publisher B.V., 1989. 137~148

5 Coquillart S. Extended free-form deformation: a sculpturing tool for 3D geometric modeling. ACM Computer Graphics (Siggraph'90), 1990,24(4):187~193

6 Karla P, Mangli A *et al*. Simulation of facial muscle actions based on rational free-form deformation. Computer Graphics Forum (Eurographics'92), 1992,2(3):59~69

7 Lamousin H, Waggenspack W. NURBS-based free-form deformation. IEEE Computer Graphics and Applications, 1994,4(9): 59~65

8 Bechmann D, Bertrand Y *et al*. Continuous free-form deformation. Computer Networks and ISDN Systems (special issue of Compugraphics'96), 1997,27(14):1715~1725

9 Coquillart S, Jancene P. Animated free-form deformations: an interactive animation technique. ACM Computer Graphics (Siggraph'91), 1991,25(4):23~26

10 Hsu W, Hughes J *et al*. Direct manipulation on free-form deformation. ACM Computer Graphics (Siggraph'92), 1992,26(2): 177~184

11 MacCracken R, Joy K. Free-Form deformation with lattice of arbitrary topology. ACM Computer Graphics (Siggaph'96), 1996,30(3):181~183

12 Moccozet L, Magnenat-Thalmann N. Dirichlet free-form deformations and their application to hand simulation. In: Thalmann N ed. Proceedings of Computer Animation'97. Geneva: IEEE Computer Society, 1997. 93~102

13 Feng Jie-qing, Ma Li-zhuang, Peng Qun-sheng. A new free-form deformation through the control of parametric surfaces. Computers and Graphics, 1996,20(4):531~539

14 Lazarus F, Coquillart S *et al*. Axial deformation: an intuitive deformation technique. Computer-Aided Design, 1994,26(8): 607~613

15 Chang Y, Rockwood A. A generalized de Casteljau approach to 3D free-form deformation. ACM Computer Graphics (Siggraph'97), 1997,30(3):187~196

16 Borrel P, Bechmann D. Deformation of N-dimensional objects. International Journal of Computational Geometry and Applications, 1991,1(4):137~155

17 Bechmann D. Multidimensional free-form deformation tools. In: de Sousa A, Hopgood B eds. State of the Art Report (Eurographics'98). Lisbon, Portugal: Blackwell Publisher, 1998. 102~110

18 Chang Geng-zhe. Bernstein polynomial via the shifting operators. American Mathematical Monthly, 1984,91(10):634~638

19 Farin G. Curves and Surfaces for Computer-Aided Geometric Design. 2nd Edition, New York: Academic Press, 1990

20 DeRose T. Compositing Bézier simplex. ACM Transactions on Graphics, 1988,7(3):198~221

21 DeRose T, Goldman R *et al*. Functional composition algorithms via blossoming. ACM Transactions on Graphics, 1993,12(3): 113~135

Accurate B-spline Free-Form Deformation of Polygonal Objects

FENG Jie-qing PENG Qun-sheng

(State Key Laboratory of CAD & CG Zhejiang University Hangzhou 310027)

Abstract Free-form deformation is an important geometric shape modification method in computer animation and geometric modeling. The authors explore it by means of functional composition via shifting operators in this paper. When the object to be deformed is represented by triangular meshes and the deformation tool is a B-spline volume, the deformed object can be accurately described as a set of triangular Bézier patches, whose degree is the sum of three directional degrees of the B-spline volume. The proposed method also solves the sample problem of free-form deformation, because free-form deformation acts on triangular meshes rather than points.

Key words Free-form deformation, functional composition, shifting operators, triangular Bézier patch.