三角形是基本的原子渲染图元。图形硬件经过调整后可以迅速变成着色片段并放入帧缓冲区。但是，在建模系统中创建的对象和动画路径可能具有许多不同的基础几何描述。曲线和曲面可以通过方程式精确描述。计算这些方程式，然后创建三角形集并将其发送到要渲染的管线下。 使用曲线和曲面的好处至少有四方面：（1）它们具有比一组三角形更紧凑的表示形式；（2）它们提供可伸缩的几何图元；（3）与直线相比，它们提供更平滑和更连续的图元（4）动画和碰撞检测可能变得更简单，更快。

紧凑的曲线表示为实时渲染提供了多个优势。首先，可以节省用于模型存储的内存（因此可以提高内存缓存效率）。这对于通常不具有PC内存的游戏机特别有用。与对表示曲面的网格进行转换相比，转换曲面通常涉及的矩阵乘法更少。如果图形硬件可以直接接受此类曲面描述，则主机CPU必须发送到图形硬件的数据量通常比发送三角形网格要少得多。

弯曲的模型描述（例如PN三角形和细分曲面）具有值得一提的特性，即可以使多边形少的模型更具说服力和更真实。各个多边形被视为曲面，因此在曲面上创建了更多顶点。较高的顶点密度可以使表面和轮廓边缘的照明更好，质量更高。有关示例，请参见图17.1。

曲面的另一个主要优点是它们是可缩放的。 曲面描述可以转换为2个三角形或2000个三角形。曲面是实时进行细节建模的自然形式：当曲面对象靠近时，更密集地采样分析表示并生成更多三角形。 对于动画，曲面的优点是需要动画的点数量少得多。 这些点可以用于形成曲面，然后可以生成平滑的镶嵌。 同样，碰撞检测可能会更有效，更准确[**939,940**]。

弯曲和曲面的主题一直是整本书的主题[**458、777、1242、1504、1847**]。我们的目标是覆盖在实时渲染中常用的曲线和曲面。

17.1 参数曲线 2020年10月12日18点27分

在本节中，我们将介绍参数曲线。 这些用于许多不同的上下文中，并使用许多不同的方法来实现。 对于实时图形，参数曲线通常用于沿预定路径移动查看器或某些对象。 这可能涉及更改位置和方向。 但是，在本章中，我们仅考虑位置路径。有关方向插补的信息，请参见第4.3.2节。 如图17.2所示，另一个用途是渲染头发。

假设您要在一定时间内将摄像机从一个点移动到另一点，而与基础硬件的性能无关。 例如，假设照相机应在一秒钟内在这些点之间移动，并且一帧的渲染需要50毫秒。 这意味着我们将能够在此期间渲染20帧。 在速度更快的计算机上，一帧可能只花费25毫秒，相当于每秒40帧，因此我们想将相机移到40个不同的位置。 查找任意一组点都可以通过参数曲线来完成。

参数曲线使用一些公式作为参数的函数来描述点.在数学上,我们将其写为,这意味着该函数为的每个值提供一个点.参数可以属于某个称为域的区间,例如.生成的点是连续的,即,然后是.松散地说,这意味着如果是一个很小的数,则和是两个非常接近的点.

在下一节中,我们将以对B´ezier曲线（参数曲线的一种常见形式）的直观和几何描述开始，然后将其置于数学设置中.然后,我们讨论了如何使用分段B´ezier曲线,并解释了曲线连续性的概念.在第17.1.4节和第17.1.5节中,我们将介绍另外两个有用的曲线,即三次Hermites和Kochanek-Bartels样条曲线.最后,我们将在第17.1.2节中介绍使用GPU渲染B´ezier曲线的方法.

17.1.1 贝塞尔曲线

线性插值绘制出一条在和两点之间的直线路径.这很简单.参见图17.3的左图.给定这些点,以下函数描述了线性插值点,其中是曲线参数，:

参数控制点在直线上的哪个位置;且给我们一个在和之间的直线上的点.因此,如果我们想在一秒钟20步内将摄像机从线性地移动到,那么我们将使用,其中是帧号（从0开始到19结束） ）.

当仅在两个点之间进行插值时,线性插值可能就足够了,但是对于路径上的更多点,通常不行.例如,当内插多个点时,连接两个线段的点(也称为关节)的突然变化变得不可接受.如图17.3的右侧所示.

为了解决这个问题，我们进一步采用线性插值的方法，并反复进行线性插值.通过这样做,我们得出了Béezier（发音为beh-zee-eh）曲线的几何构造.作为历史记录,Béezier曲线是由Paul de Casteljau和Pierre B´ezier独立开发的,用于法国汽车工业.之所以称它们为B´ezier曲线,是因为即使de Casteljau在B´ezier之前发表了他的技术报告[458]，B´ezier仍能在de Casteljau之前公开发表他的研究.

首先,要能够重复插值,我们必须添加更多点.例如,可以使用称为控制点的三个点和.假设我们要找到,即的曲线上的点.我们使用,通过线性插值从和计算两个新点和.参见图17.4.最后,我们使用从和通过线性插值计算.我们定义.使用这种技术,我们得到以下关系:

这是一个抛物线,因为最大t的最大次数为2.实际上,在给定个控制点的情况下,曲线的度数为n.这意味着更多的控制点为曲线提供了更多的自由度。 第一级曲线是直线（称为线性），第二级曲线称为二次曲线，第三级曲线称为三次曲线，第四级曲线称为四次曲线，依此类推.

这种重复或递归的线性插值通常称为de Casteljau算法[458,777].使用1个控制点时的外观示例如图17.5所示,为了概括起见,而不是如本示例中那样使用点,而是使用以下表示法.控制点表示为,因此在示例中,和.然后,在已经进行了次线性插值之后,获得了中间控制点.在我们的示例中我们有和.可以使用以下所示的递归公式来描述个控制点的B´ezier曲线,其中是初始控制点:

注意,曲线上的一个点由描述.这并不像看起来那么复杂.再次考虑当我们从和的三个点(分别等于和)构造B´ezier曲线时会发生什么.三个控制点意味着.有时“(t)”会从处删除.在第一步中,,得到,而.最后,对于,我们得到,这与对的要求相同.图17.6给出了一般情况下工作原理的说明.

现在我们已经掌握了Béezier曲线如何工作的基础知识,现在我们可以看一下对相同曲线的更数学描述.

使用伯恩斯坦多项式的贝塞尔曲线 2020年10月13日18点38分

如公式17.2所示,可以使用代数公式来描述二次B´ezier曲线.事实证明,每个B´ezier曲线都可以用这样的代数公式来描述,这意味着您无需进行重复插值.这在下面的公式17.4中显示,得出的曲线与公式17.3中所述相同.贝塞尔曲线的这种描述称为伯恩斯坦形式:

该函数包含伯恩斯坦多项式,有时也称为B´ezier基函数,

此方程式中的第一项二项式系数在第1章的方程式1.6中定义.Bernstein多项式的两个基本属性如下:

第一个公式表示当也是从0到1时，Bernstein多项式处于0到1的区间内.第二个公式表示等式17.4中的所有Bernstein多项式项对于曲线的所有不同阶数之和等于1（这是 见图17.7）。 宽松地说，这意味着曲线将保持“接近”控制点.实际上,整个B´ezier曲线将位于控制点的凸包中(请参见我们的在线线性代数附录),这可根据公式17.4和17.6得出.在计算曲线的边界面积或体积时,这是一个有用的属性.有关示例,请参见图17.5.

在图17.7中,显示了和的伯恩斯坦多项式.这些也称为混合函数.(线性插值)的情况是说明性的,因为它表示曲线和.这意味着,当时,则,并且当增加时,的混合权重减小,而的混合权重增加相同的数量,并且权重的总和等于1.最后,当时,.通常,对于所有B´ezier曲线,且成立,也就是说,端点是被插值的(即在曲线上).的确,该曲线在时与向量相切,在时与相切.另一个有用的特性是,代替了计算B´ezier曲线上的点,然后旋转曲线时,可以先旋转控制点,然后可以计算曲线上的点.通常,控制点少于曲线上生成的点,因此先转换控制点效率更高.

作为有关B´ezier曲线的Bernstein版本如何工作的示例,假设,即二次曲线.等式17.4为

与公式17.2相同.请注意,上面的混合函数是显示在图17.7中间的函数.同样,三次曲线简化为

该方程可以重写为矩阵形式

这在进行数学简化时很有用.

通过在公式17.4中收集tk形式的项,可以看出,每条B´ezier曲线都可以用以下形式写成，称为幂形式,其中是通过收集项而掉落的点:

为了得到B´ezier曲线的导数,可以很容易地对公式17.4进行微分.重组和收集项后,结果显示在[458]中:

实际上,导数也是Bezier曲线,但比低1度.

B´ezier曲线的潜在缺点是它们不会通过所有控制点(端点除外).另一个问题是,程度随着控制点的数量而增加,从而使评估越来越昂贵.一种解决方案是在每对后续控制点之间使用一条简单的低度曲线,并确保这种分段插值具有足够高的连续性.这是第17.1.3-17.1.5节的主题.

实贝塞尔曲线

虽然B´ezier曲线可用于许多事情,但它们没有那么多的自由度,只能自由选择控制点的位置.同样,并非所有曲线都可以用B´ezier曲线来描述.例如,圆通常被认为是简单形状,但是不能由一个或一组B´ezier曲线定义.一种选择是有理Béezier曲线.此类曲线由公式17.12中所示的公式描述:

分母是伯恩斯坦多项式的加权和，而分子是标准B´ezier曲线的加权形式（公式17.4）.对于这种类型的曲线,用户拥有权重作为附加的自由度.有关这些曲线的更多信息,请参见Hoschek和Lasser的[777]和Farin的书[458].Farin还描述了如何用三个有理B´ezier曲线描述一个圆.

17.1.2 GPU上的有界贝塞尔曲线 2020年10月14日19点07分

将介绍一种在GPU上渲染B´ezier曲线的方法[**1068,1069**].具体来说,目标是“有界Béezier曲线”,其中曲线之间的区域以及第一个和最后一个控制点之间的直线被填充.通过使用专用像素着色器渲染三角形,可以实现一种非常简单的方法.

我们使用具有控制点和的二次曲线,即二阶B´ezier曲线.如果将这些顶点的纹理坐标设置为和,则在渲染三角形时将照常插值纹理坐标.我们还在每个像素的三角形内评估以下标量函数,其中和是插值的纹理坐标:

然后,像素着色器确定像素是在内部（），还是在外部.如图17.8所示.使用此像素着色器渲染透视投影的三角形时,我们将获得相应的投影的B´ezier曲线.Loop和Blinn [1068,1069]给出了一个证明.

例如，这种技术可用于呈现TrueType字体.如图17.9所示.Loop和Blinn还显示了如何渲染有理二次曲线和三次曲线,以及如何使用此表示进行抗锯齿.由于文本渲染的重要性,该领域的研究仍在继续.有关相关算法,请参见第15.5节.

17.1.3 连续性和分段贝塞尔曲线 2020年10月14日19点40分

假设我们有两条三次的B´ezier曲线,即分别由四个控制点定义.第一条曲线由定义,第二条曲线由定义,.要合并这些曲线,我们可以设置.这一点称为关节.但是,如图17.10所示,使用此简单技术将使关节不平滑.由多个曲线段(在这种情况下为两个)形成的合成曲线称为分段B´ezier曲线,在此表示为.此外,假设我们想要，以及.因此,当我们到达和的时间分别为和.有关符号,请参见图17.10.从上一节中我们知道,为定义了一条Béezier曲线，因此对于由定义的第一个曲线段来说,这是可行的,因为处的时间为0,处的时间为1.但是,当时会怎样？答案很简单:我们必须使用第二条曲线段,然后将参数间隔从转换并缩放为[0，1].使用以下公式完成此操作:

因此，将'馈入由定义的贝塞尔曲线段中.这很容易概括为将多个Bezier曲线缝合在一起.

连接这些曲线的一种更好的方法是利用这样一个事实,即在Béezier曲线的第一个控制点处,切线平行于(第17.1.1节).同样,在最后一个控制点,三次曲线与相切.这种行为可以在图17.5中看到.因此,为使两条曲线在切线处相切连接,第一条和第二条曲线的切线应在此处平行.更正式地说,以下内容应成立:

这仅表示关节处的传入切线应该与传出切线的方向相同.

通过在公式17.15中使用公式17.16[458]定义的,可以实现甚至更好的连续性:

这也显示在图17.10中.如果改为设置,则,因此,当每个曲线段上的时间间隔相等时,传入和传出切线向量应相同.但是,这在时不起作用.曲线看起来将相同,但是在复合曲线上移动的速度将不平滑.公式17.16中的常数可以解决这一问题.

使用分段曲线的一些优点是可以使用较低度的曲线，并且生成的曲线将经过一组点。在上面的例子中，对于两个曲线段中的每一个，使用度数为3，即立方。经常使用三次曲线，因为它们是可以描述S形曲线（称为拐点）的最低度曲线。所得曲线p（t）进行插值，即穿过点q0，q3 = r0和r3。

在这一点上，通过示例介绍了两个重要的连续性度量.接下来是曲线的连续性概念的数学表述。对于一般的曲线，我们使用Cn表示法来区分关节处不同类型的连续性。这意味着所有n个一阶导数在整个曲线上应该是连续且非零的。 C0的连续性意味着该段应在同一点连接，因此线性插值满足此条件。本部分的第一个示例就是这种情况。 C1的连续性意味着，如果我们在曲线上的任何一点（包括关节）导出一次，结果也应该是连续的。本节的第三个示例就是这种情况，其中使用了公式17.16。

还有一个度量值表示为Gn。 让我们以G1（几何）连续性为例。 为此，在关节处相遇的曲线段的切线矢量应平行且方向相同，但长度无关。 换句话说，G1的连续性比C1弱，并且C1的曲线始终是G1，除非两条曲线的速度在两条曲线的交汇点变为零且在交汇之前它们的切线不同。 几何连续性的概念可以扩展到更高的尺寸。 图17.10的中间图显示了G1连续性。

17.1.4 立方Hermite插值 2020年10月15日19点31分

贝兹曲线适合描述平滑曲线背后的理论,但有时无法预测.在本节中,我们将介绍三次Hermite插值,并且这些曲线往往更易于控制.原因是,三次曲线Hermite曲线不是由四个控制点来描述三次B´ezier曲线,而是由起点和终点和以及起点和终点切线和定义. Hermite插值,其中,为

**17.1.5 Kochanek-Bartels曲线** 2020年10月15日19点41分

在两个以上的点之间进行插值时,可以连接多个Hermite曲线.但是,这样做时,选择提供不同特性的共享切线会有自由度.在这里,我们将介绍一种计算此类切线的方法,称为Kochanek-Bartels曲线.假设我们有个点,,应插入个Hermite曲线段.我们假设在每个点只有一个切线,并且我们开始看“内”切线.可以将处的切线计算为两个和弦[917]:和的组合,如图17.13左侧所示.

首先,引入张力参数,该参数修改切线向量的长度.这控制了曲线在关节处的锐度.切线计算为

图17.13右侧的第一行显示了不同的张力参数.默认值为;较高的值会给出更锐利的弯曲(如果,则在关节处会出现一个环),而负值会使在关节附近的绷紧曲线更少.其次,引入一个影响切线方向(间接影响切线长度)的偏移参数.同时利用张力和偏移

其中默认值为.正偏向的弯曲更指向和弦,负偏向的弯曲更指向另一个和弦:.这在图17.13的右下一行中显示.用户既可以设置张力和偏置参数,也可以让它们具有默认值,这会产生通常称为Catmull-Rom样条曲线的信息[236].也可以使用这些公式计算第一个和最后一个点的切线,其中一个和弦的长度简单设置为零.

可以将控制关节行为的另一个参数并入切线方程[**917**].但是,这需要在每个关节处引入两个切线,一个切线表示为(对于源),一个切线表示为(对于目标).参见图17.14.注意,和之间的曲线段使用切线和.切线计算如下,其中是连续性参数:

同样,是默认值,这使.设置给出,并且,在接头处产生一个尖角,仅为.增加的值会使和越来越相似.对于,则.当达到时，我们得到以及.因此,连续性参数是向用户提供更多控制的另一种方式,并且如果需要的话,它可以使关节处出现尖角.

张力,偏移和连续性的组合(默认参数值为)为

仅当所有曲线段都使用相同的时间间隔长度时,方程17.20和17.22才起作用.为了说明曲线段的不同时间长度,必须像第17.1.3节中那样调整切线.调整后的切线表示为和,

其中.

17.1.6 B样条 2020年10月16日11点46分

在这里,我们将简要介绍B样条的主题,并且我们将特别关注立方均匀B样条.通常,B样条曲线与B´ezier曲线非常相似,可以表示为(使用位移基函数),(由控制点加权)和的函数:

在这种情况下,这是一条曲线,其中是x轴,是轴,控制点只是均匀分布的值.要获得更广泛的报道,请参见杀手B的著作[111],法林[458]以及霍舍克和拉塞尔的著作[777].

在这里,我们将遵循Ruijters等人[1518]的介绍,并介绍均匀三次B样条的特殊情况.基本函数由三部分缝合在一起:

图17.15左侧显示了此基本函数的构造.此函数在任何地方都具有连续性,这意味着如果将多个B样条曲线段缝合在一起,则合成曲线也将为.三次曲线具有连续性,通常,度为的曲线具有连续性.通常,基本函数如下创建.是一个“平方”函数，即如果,它为1,如果,它为0.5,否则为0.下一个基础函数是通过对积分来创建的,这为我们提供了一个帐篷函数.之后,通过对进行积分来创建基函数,该函数给出了更平滑的函数,即.重复此过程以获得,依此类推.

图17.15右侧说明了如何评估曲线段,其公式为

请注意,任何时候都只会使用四个控制点,这意味着曲线具有局部支持,即需要有限数量的控制点.使用定义函数为

Ruijters等人[1518]表明,这些可以重写

在图17.16中,我们显示了将两条均匀的三次B样条曲线缝合在一起的结果.一个主要优点是曲线是连续的,具有与基函数相同的连续性,在三次B样条曲线的情况下为.从图中可以看出,不能保证曲线会通过任何控制点.请注意,我们还可以为坐标创建B样条曲线,这将在平面中给出一条普通曲线（而不仅仅是一个函数）.这样得到的二维点将是,即简单的等式17.26的两种不同评估,一个是x,另一个是y.

我们已经展示了如何仅使用均匀的B样条曲线.如果控制点之间的间距不均匀,则方程将变得更加复杂,但更加灵活[**111,458,777**].

17.2 参数曲线曲面 2020年10月23日16点35分

参数曲线的自然扩展是参数曲面.打个比方,三角形或多边形是线段的扩展,我们从一维到二维.参数曲面可用于对具有曲面的对象进行建模.参数化曲面由少量控制点定义.参数化曲面的细分是评估多个位置上的曲面表示并将它们连接起来以形成近似真实曲面的三角形的过程.这样做是因为图形硬件可以有效地渲染三角形.然后在运行时,可以将曲面细分为所需的任意多个三角形.因此,参数曲面非常适合在质量和速度之间进行权衡,因为更多的三角形需要更多的渲染时间,但着色和轮廓更好.参数化曲面的另一个优点是可以对控制点进行动画处理,然后可以对曲面进行细分.这与直接为大三角形网格设置动画相反,后者可能会更加昂贵.

本节首先介绍Béezier面片,这些面是具有矩形域的曲面.这些也称为张量积Béezier曲面.然后给出具有三角形区域的B´ezier三角形,然后在17.2.3节中讨论连续性.在第17.2.4节和第17.2.5节中,提出了两种方法,分别用B´ezier三角形替换每个输入三角形.这些技术分别称为PN三角形和Phong镶嵌.最后,第17.2.6节介绍了B样条补丁.

17.2.1 贝塞尔曲面

在17.1.1节中介绍的B´ezier曲线的概念可以从使用一个参数扩展到使用两个参数,从而形成曲面而不是曲线.让我们开始将线性插值扩展到双线性插值.现在,我们不仅使用两个点,还使用四个点,分别称为和,如图17.17所示.现在,我们使用两个参数,而不是使用单个参数.使用线性内插和得出和:

接下来,使用在另一个方向上线性插值点和.这将产生双线性插值:

请注意,这与用于纹理映射的双线性插值使用的方程类型相同(第179页的方程6.1).公式17.30描述了最简单的非平面参数化曲面,其中使用的不同值生成曲面上的不同点.该域,即一组有效值,是,这意味着和都应属于.当域为矩形时,生成的表面通常称为面片.

为了通过线性插值法扩展B´ezier曲线,添加了更多点并重复了插值法.相同的策略可以用于补丁.假设使用以3×3网格排列的九个点.如图17.18所示,其中也显示了符号.要从这些点形成一个双二次Béezier面片,我们首先需要双线性插值四次以创建四个中间点,如图17.18所示.接下来,根据先前创建的点对表面上的最终点进行双线性插值.

上述重复的双线性插值是de Casteljau算法对面片的扩展.在这一点上,我们需要定义一些符号.曲面的度为.控制点是,其中和属于.因此,个控制点用于度数为n的补丁.请注意,控制点应以零作为上标,即,但这通常会被省略，并且有时在没有混乱的情况下,我们使用下标代替.以下等式描述了使用de Casteljau算法的B´ezier补丁:

其中.

与B´ezier曲线相似，B´ezier面片上的点为.也可以使用Bernstein多项式以Bernstein形式描述B´ezier面片,如公式17.32所示:

注意,在公式17.32中,有两个参数和用于表示表面的度.有时将“复合”度表示为.大多数情况下,,这简化了实现.例如,的结果是首先对次进行双线性插值,然后对次进行线性插值.如图17.19所示.通过将等式17.32重写为式,可以找到另一种解释

其中.如方程式17.33的最下面一行所示,当我们固定值时,这只是一条B´ezier曲线.假设，则可以从B´ezier曲线计算出点,然后对于,公式17.33描述了B´ezier曲面上的B´ezier曲线.

接下来,将介绍B´ezier补丁的一些有用属性.在公式17.32中通过设置和,很容易证明B´ezier面片会插值,即经过角控制点和.同样,斑块的每个边界都由边界上的控制点形成的度为的B'ezier曲线来描述.因此,拐角控制点处的切线由这些边界B´ezier曲线定义.每个角控制点都有两个切线,分别在方向和方向上一个.与B´ezier曲线一样,面片也位于其控制点的凸包内,并且

其中.最后,旋转控制点然后在补丁上生成点在数学上与(在生成速度上快于)在补丁上生成点然后旋转这些点相同.

对方程17.32进行偏微分可得出[458]以下方程:

可以看出,patch的度数在可微的方向上减少了一个.然后将未归一化的法向向量的形式为

在图17.20中,显示了控制网格以及实际的B'ezier面片.移动控制点的效果如图17.21所示.

实贝塞尔Patches 2020年10月29日17点06分

正如B´ezier曲线可以扩展为有理B´ezier曲线(第17.1.1节),从而引入更多的自由度一样,B´ezier面片也可以扩展为有理B´ezier面片:

有关此类修补程序的信息，请查阅Farin的书[458]和Hochek和Lasser的书[777].同样,有理B´ezier三角形是B´ezier三角形的扩展,接下来处理.

17.2.2 贝塞尔三角形 2020年10月29日17点14分

尽管通常将三角形视为比矩形更简单的几何图元,但在Béezier曲面上却并非如此：B´ezier三角形并不像B´ezier面片那么简单.这种贴片值得一提,因为它可用于快速简便地形成PN三角形和用于Phong镶嵌.请注意,某些游戏引擎(例如,虚幻引擎,Unity和Lumberyard)支持Phong细分和PN三角形.

控制点位于三角形网格中,如图17.22所示.B´ezier三角形的度数为,这意味着每边有个控制点.这些控制点表示为,有时缩写为.注意,对于所有控制点,,并且.因此,控制点的总数为

B´ezier三角形也基于重复插值就不足为奇了.但是,由于域的三角形形状,必须使用质心坐标(第22.8节)进行插值.回想一下三角形中的一个点可以描述为,其中是质心坐标.对于三角形内的点,必须满足以下条件:,并且.基于此，B´ezier三角形的de Casteljau算法为

其中.

在的B´ezier三角形上的终点是.伯恩斯坦形式的B´ezier三角形形式是

伯恩斯坦多项式现在同时取决于和,因此计算方式有所不同,如下所示:

偏导数为[475]

B´ezier三角形的一些不足为奇的特性是它们可以插值(通过)三个角控制点,并且每个边界都是该边界上的控制点所描述的B´ezier曲线.而且,这些表面位于控制点的凸包中.贝兹埃三角形如图17.23所示.

17.2.3 连续性 2020年10月29日17点42分

当从Bezier曲面构造复杂的对象时,人们通常希望将多个不同的B´ezier曲面缝合在一起以形成一个复合曲面.为了获得良好的外观效果,必须注意确保在整个表面上获得合理的连续性. 这与第17.1.3节中曲线的精神相同.

假设应将两个三次Béezier贴片拼凑在一起.它们每个都有4×4个控制点.如图17.24所示,对于,左边的色块具有控制点,右边的色块具有控制点.为确保连续性,Patch必须在边界处共享相同的控制点,即.

但是,这不足以获得美观的复合表面.取而代之的是,将介绍一种提供连续性的简单技术[458].为此,我们必须限制最靠近共享控制点的两行控制点的位置.这些行是和.对于每个,点和必须是共线的,也就是说,它们必须位于一条线上.此外,它们必须具有相同的比率,这意味着.在此,是一个常数，并且对于所有都必须相同.示例如图17.24和17.25所示.

这种构造消耗了设置控制点的许多自由度.将四个补丁缝在一起,共享一个共同的角时,可以更加清楚地看到这一点.该结构如图17.26所示.结果显示在该图的右侧,其中显示了共享控制点周围的八个控制点的位置.这九个点必须全部位于同一平面上,并且必须形成双线性面,如图17.17所示.如果对拐角处(且仅在拐角处)的连续性感到满意,则使这9个点共面即可.这使用较少的自由度.

B´ezier三角形的连续性通常更为复杂,如果约束B´ezier面片和三角形的条件[458，777].当构造包含许多B´ezier曲面的复杂对象时,通常很难看到在所有边界上都获得了合理的连续性.一种解决方法是转向细分表面,在17.5节中进行了处理.

请注意,连续性是跨边界的美观纹理所必需的.对于反射和着色,使用G1连续性可以获得合理的结果.或更高给出更好的结果.图17.25显示了一个示例.

在以下两个小节中，我们将介绍两种利用三角形顶点处的法线来为每个输入(平坦)三角形得出B´ezier三角形的方法.

17.2.4 PN三角形 2020年10月29日18点06分

给定一个在每个顶点具有法线的输入三角形网格,Vlachos等人[1819]的PN三角形方案的目标是与仅使用三角形相比,构造一个外观更好的表面.字母“PN”是“点与法线”的缩写,因为这是生成曲面所需的所有数据.它们也称为N补丁.该方案试图通过创建曲面来替换每个三角形来改善三角形网格的着色和轮廓.细分硬件能够动态制作每个曲面,因为细分是根据每个三角形的点和法线生成的,不需要邻居信息.有关示例,请参见图17.27.这里介绍的算法是基于van Overveld和Wyvill [1341]的工作.

假设我们有一个三角形，其顶点和的法线分别为和.基本思想是使用此信息为每个原始三角形创建三次B´ezier三角形,并根据我们希望的数量从B´ezier三角形中生成尽可能多的三角形.

为了缩短表示法,将使用.三次Béezier三角形的计算公式为

参见图17.22.为了确保两个PN三角形之间的边界处的连续性,可以根据拐角控制点和这些拐角处的法线来确定边缘上的控制点.(假设法线在相邻三角形之间共享).

假设我们要使用控制点和的法线计算,如图17.28所示.只需将点沿法线的方向投影到由和的定义的切平面上.假设归一化的法线,则点计算为

可以类似地计算其他边界控制点,因此只留下计算内部控制点.如下面的方程式所示,该操作遵循二次多项式[457,458]:

Vlachos等人[1819]并未使用公式17.42计算表面上的两个切线以及随后的法线，而是选择使用二次方案对法线进行插值,如下所示:

可以将其视为度数为2的B´ezier三角形,其中控制点是六个不同的法线.在公式17.46中,度数的选择(即二次数)是很自然的,因为导数比实际的B´ezier三角低1度,并且因为法线的线性插值不能描述拐点.参见图17.29.

为了能够使用公式17.46，需要计算法线控制点和.一种直观但有缺陷的解决方案是使用和的平均值(原始三角形顶点处的法线)来计算.但是,当时,将再次遇到图17.29左下方所示的问题.取而代之的是,首先通过取和的平均值,然后在平面中反映该法线来构造,如图17.30所示.该平面的法线平行于端点和之间的差.由于只有法线向量会在中反射,因此我们可以假定通过原点,因为法线与平面上的位置无关.另外,请注意,每个法线均应标准化.数学上,的非标准化版本表示为[1819]

最初,van Overveld和Wyvill在此等式中使用的系数为,而不是.通过查看图像很难判断哪个值最好,但是使用可以很好地解释平面中的真实反射.

此时,已经计算出三次Béezier三角形的所有B´ezier点以及用于二次插值的所有法线向量.仅需在B´ezier三角形上创建三角形,以便可以渲染它们.这种方法的优点是,表面具有更好的轮廓和形状,而成本相对较低.

指定详细程度的一种方法如下.原始三角形数据被认为是LOD0.然后,LOD数随三角形边缘上新引入的顶点数而增加.因此,LOD 1在每个边上引入一个新顶点,从而在B´ezier三角形上创建四个子三角形,而LOD 2在每个边上引入两个新顶点,生成9个子三角形.通常,LOD n生成个子三角形.为了防止在Béezier三角形之间出现裂纹,必须以相同的LOD细分网格中的每个三角形.这是一个严重的缺点,因为将与大三角形一样细分一个小三角形.可以使用诸如自适应曲面细分(第17.6.2节)和分数曲面细分(第17.6.1节)之类的技术来避免这些问题。

PN三角形的一个问题是折痕难以控制,并且经常需要在所需折痕附近插入额外的三角形.Béezier三角形之间的连续性仅为,但在许多情况下它们仍然可以接受.这主要是因为法线在三角形之间是连续的,因此一组PN三角形模拟了曲面.Boubekeur等人[**181**]提出了一种更好的解决方案,其中一个顶点可以具有两个法线,并且两个这样的相连顶点会产生折痕.请注意,要获得美观的纹理,三角形(或面片)之间的边界需要连续性.还应该知道的是,如果两个相邻的三角形不共享相同的法线,则会出现裂纹.Grun [**614**]描述了一种进一步改善PN三角形连续性质量的技术.Dyken等人[**401**]提出了一种受PN三角形启发的技术,其中只有从观看者处看到的轮廓才被自适应地细分,因此变得更加弯曲.这些轮廓曲线的生成方式与PN三角形曲线类似.为了获得平滑的过渡,它们在粗糙轮廓和棋盘格轮廓之间融合.为了提高连续性,Funfzig等人[**505**]提出了PNG1三角形,这是对PN三角形的一种修改,这些PN三角形到处都有连续性.McDonald和Kilgard [**1164**]提出了PN三角形的另一种扩展,它可以处理相邻三角形上的不同法线.

17.2.5 Phong细分 2020年10月29日19点31分

Boubekeur和Alexa [**182**]提出了一种称为Phong棋盘格化的表面构造,它与PN三角形有很多相似之处,但评估速度更快,更易于实现.称基本三角形的顶点为和,并使对应的归一化法线为和.首先,回想一下,重心坐标上的基本三角形上的一个点计算为

在Phong着色中,也使用上面的方程在法线三角形上插入法线,但点被法线替换.Phong细分尝试使用重复插值来创建Phong着色法线插值的几何版本,从而产生B´ezier三角形.对于此讨论,我们将参考图17.31.第一步是创建一个函数,将点在基三角形上投影到由点和法线定义的切平面.这样做是

代替使用三角形顶点执行线性插值(公式17.48),使用函数完成线性插值,结果是

为了增加灵活性，添加了一个形状因子,该形状因子插在基础三角形和公式17.50之间,从而得出Phong细分的最终公式:

建议将设置为[182].生成此曲面所需的唯一信息是基本三角形的顶点和法线以及用户提供的,这可以快速评估此曲面.所得的三角形路径是二次的,即,其度数低于PN三角形.就像在标准Phong着色中一样,简单地对法线进行线性插值.有关说明将“Phong细分”应用于网格的效果的示例,请参见图17.32.

17.2.6 B样条曲面 2020年10月29日19点33分

17.1.6节简要介绍了B样条曲线,这里我们将介绍B样条曲面.可以将第732页上的公式17.24推广为Bspline补丁

这与B´ezier补丁公式(公式17.32)非常相似.请注意,是表面上的三维点.如果将此函数用于纹理过滤,则公式17.52将是一个高度场,而将是一维,即高度.

对于双三次B样条斑块,公式17.25中将使用公式17.25中的函数.总共需要4×4个控制点,等式17.52描述的实际表面斑块将位于最里面的2×2个控制点内.如图17.33所示. 注意,双三次B样条曲线斑块对于Catmull-Clark细分曲面也是必不可少的(第17.5.2节).有很多不错的书,其中有关于B样条曲面的更多信息[111,458,777].

17.3 隐式曲面 2020年10月29日19点53分

至此,仅讨论了参数曲线和曲面.隐式曲面构成了另一个用于表示模型的方法.无需使用某些参数(例如和)来明确描述表面上的点,而是使用以下形式(称为隐式函数):

解释如下:如果将点代入隐函数中,结果为零,则点在隐表面上.隐式曲面通常用于与射线的相交测试中(第22.6-22.9节),因为与相应的参数化曲面(如果有)相比,它们更容易相交.隐式曲面的另一个优点是,可以很容易地对它们应用构造性实体几何算法,也就是说,可以将彼此进行逻辑“与”或“或”运算以相互减去对象.同样,物体可以很容易地融合和变形.

位于原点的隐式曲面的一些示例是

这些值得解释.球体就是从到原点的距离减去半径,因此,如果在半径为的球体上,则等于0.否则,将返回一个有符号距离,其中负数表示在球体内部,而正数在球体外部.因此,这些函数有时也称为有符号距离函数(SDF).平面只是的坐标,即轴为正的一侧.对于圆角矩形的表达式,我们假设按每个分量计算绝对值()和向量的最大值.同样,是盒子半边的向量.参见图17.34中的圆形框.标题中说明了该公式.要获得非圆角矩形,只需将设置为0.

隐式曲面的法线由偏导描述,称为梯度,记作:

为了能够准确地对其进行评估,公式17.55中的必须是可微的,因此也必须是连续的.在实践中,通常使用一种称为中心差分的数值技术,该技术使用场景函数[**495**]进行采样:

和也是如此.回想和,并且是一个小数.

为了用公式17.54中的图元构建场景,使用了联合运算符.例如,是一个由球体和平面组成的场景.联合运算符通过采用其两个操作数中的最小值来实现,因为我们想找到最接近的曲面.平移是通过在调用有符号距离函数之前平移来完成的,即是一个由平移的球体.旋转和其他变换可以相同的精神完成,即,将逆变换应用于.通过使用而不是作为有符号距离函数的参数,可以在整个空间上重复对象.

隐式曲面的混合是一个很好的功能,可以用于通常被称为blobby建模[**161**],软物体或metaball[**67,558**]的功能.有关示例,请参见图17.35.基本思想是使用几个简单的图元,例如球体,椭球或任何可用的图元,并将它们平滑地混合.每个对象都可以看作是一个原子,并且在混合后获得原子分子.混合可以通过许多不同的方式完成.将两个距离和使用半径的常用方法[**1189,1450**]是

其中是混合距离.虽然此函数仅将最短距离混合到两个对象,但可以重复使用该函数以混合更多对象(请参见图17.35的右侧).

为了可视化一组隐式函数,通常使用的方法是光线行进[**673**].一旦可以对场景进行光线行进,就可以生成阴影,反射,环境光遮挡和其他效果.图17.36说明了在有符号距离场内的光线行进.在射线的第一点,我们评估到场景的最短距离.由于这表示在半径为的周围有一个球体,没有其他物体在附近,因此我们可以沿射线方向移动射线单位,依此类推,直到到达某个epsilon内的表面,或者当预定义的射线行进步骤已经完成,在这种情况下,我们可以假定背景已经达到.图17.37给出了两个很好的例子.

每个隐式曲面也可以变成由三角形组成的曲面.有几种算法可用于执行此操作[**67,558**].一个著名的例子是marching cubes算法,在13.10节中介绍.网络上有Wyvill和Bloomenthal使用算法执行多边形化的代码[**171**],以及de Ara´ujo等人[**67**]提出了对隐式表面多边形化的最新技术的研究.Tatarchuk和Shopf [**1744**]描述了一种称为行进四面体的技术,其中GPU可用于在三维数据集中查找等值面.第48页的图3.13显示了使用几何体着色器提取等值面的示例.肖等[**1936**]提出了一种流体模拟系统,其中GPU计算100k粒子的位置并使用它们显示等值面,所有这些都以交互速率进行.

17.4 细分曲线 2020年11月5日11点02分

细分技术用于创建平滑的曲线和曲面.之所以将它们用于建模,是因为它们可以弥合离散表面(三角形网格)和连续表面(例如,一系列B´ezier面片)之间的间隙,因此可以用于详细程度的技术(第19.9节).在这里,我们将首先描述细分曲线的工作方式,然后讨论更流行的细分曲面方案.

通过使用**拐角切割**的示例可以最好地解释细分曲线.参见图17.38.最左边的多边形的角被切除,从而创建一个新的多边形,其顶点数量是原来的两倍.然后,将这个新多边形的角切掉,依此类推,直到无穷大(或更确切地说,直到我们看不到任何差异为止).由于所有角都被切除,因此生成的曲线称为极限曲线,该曲线是平滑的.由于去除了所有尖角(高频),因此该过程也可以被视为低通滤波器.该过程通常写为,其中是起始多边形,也称为控制多边形,而是极限曲线.

该细分过程可以通过许多不同的方式来完成,每种方法都具有细分方案.图17.38中所示的方法称为Chaikin方案[**246**],其工作原理如下.假设多边形的个顶点为,其中上标表示细分级别.Chaikin方案在原始多边形的每个后续顶点对(例如和)之间创建了两个新顶点,

可以看出,上标从变为,这意味着我们从一个细分级别转到下一个细分级别,即.在执行这样的细分步骤之后,将丢弃原始顶点,并重新连接新的点.在图17.38中可以看到这种行为,其中新点的创建是从原始顶点向相邻顶点的1/4处.细分方案的优点在于快速生成平滑曲线的简单性.但是,虽然可以证明Chaikin的算法会生成二次B样条曲线[111,458,777,1847],但并没有立即获得17.1节中曲线的参数形式.到目前为止,提出的方案适用于(封闭的)多边形,但是大多数方案也可以扩展为适用于开放式折线.对于Chaikin,唯一的区别是折线的两个端点保留在每个细分步骤中(而不是被丢弃).这使曲线穿过端点.

细分方案有两种不同的类别,即近似和插值.Chaikin的方案是近似值,因为极限曲线通常不位于初始多边形的顶点上.这是因为顶点被丢弃(或对于某些方案是更新的).相反,插值方案保留了上一个细分步骤中的所有点,因此极限曲线遍历了等所有点.这意味着方案会插值初始多边形.使用与图17.38相同的多边形的示例如图17.39所示.该方案使用四个最近的点来创建新点[**402**]:

公式17.59中的第一行仅表示我们保留了上一步骤中的点而不对其进行更改(即插值),而第二行用于在和之间创建一个新点.权重称为**张力参数[tension parameter]**.当时,是线性插值的结果,但是当时,我们得到的行为如图17.39所示.可以显示[402],当时,结果曲线为.对于开放式折线,在端点处会遇到问题,因为在新点的两边都需要两个点,而我们只有一个.如果端点旁边的点反映在端点上,则可以解决此问题.因此,对于折线的起点,会在上反射以获得.然后将这一点用于细分过程.的创建如图17.40所示.

另一种近似方案使用以下细分规则:

第一行更新现有点,第二行计算两个相邻点之间的线段上的中点.该方案生成三次B-spline曲线(第17.1.6节).有关这些曲线的更多信息,请查阅SIGGRAPH关于细分的课程(**1977**),Killer B的书籍[111],Warren and Weimer的细分书籍[**1847**]或Farin的CAGD书籍[**458**].

给定点及其相邻点,可以直接将该点“推”到极限曲线,即确定在p∞上的坐标.切线也是可能的.例如,请参见Joy对此主题的在线介绍[**843**].

细分曲线的许多概念也适用于细分曲面,下面将进行介绍.

17.5 细分曲面 2020年11月5日12点12分

细分曲面是一种强大的范例,用于从具有任意拓扑的网格定义平滑,连续,无裂纹的曲面.与本章中的所有其他曲面一样,细分曲面也提供了无限的细节层次.也就是说,您可以根据需要生成任意数量的三角形或多边形,并且原始表面表示很紧凑.图17.41显示了细分表面的示例.另一个优点是细分规则简单易行.缺点是表面连续性的分析通常涉及数学.但是,这种分析通常仅对那些希望创建新的细分方案的用户感兴趣,而这超出了本书的范围.有关此类详细信息,请查阅Warren和Weimer的书[**1847**]和有关细分的SIGGRAPH课程[**1977**].

通常,可以将表面(和曲线)的细分视为两阶段过程[**915**].从称为控制网格或**控制笼[control cage]**的多边形网格开始，第一个阶段称为**细化阶段[refinement phase]**,它创建新的顶点并重新连接以创建新的较小的三角形.第二个步骤称为平滑阶段,通常计算网格中某些或所有顶点的新位置.如图17.42所示.这两个阶段的细节是细分方案的特征.在第一个阶段中,可以用不同的方式分割多边形,在第二个阶段中,细分规则的选择具有不同的特性,例如连续性级别,曲面是近似还是插值,这是17.4节中介绍的属性.

细分方案的特征是固定的或非固定的,均匀的或不均匀的,以及它是基于三角形的还是基于多边形的.固定方案在每个细分步骤中都使用相同的细分规则,而非平稳方案可以根据当前正在处理的步骤来更改规则.下面处理的方案都是固定的.均匀方案对每个顶点或边使用相同的规则,而非均匀方案可能对不同的顶点或边使用不同的规则.例如,通常将一组不同的规则用于曲面边界上的边缘.基于三角形的方案仅对三角形起作用,因此仅生成三角形,而基于多边形的方案对任意多边形起作用.

接下来介绍几种不同的细分方案.在此之后,提出了两种扩展细分表面用途的技术以及细分法线,纹理坐标和颜色的方法.最后,提出了一些实用的细分和渲染算法.

17.5.1 Loop细分 2020年11月5日12点23分

Loop的方法[**767,1067**]是三角形的第一个细分方案.就像第17.4节中的最后一个方案一样,它是近似的,它会更新每个现有的顶点并为每个边创建一个新的顶点,此方案的连接性如图17.43所示,可以看出,每个三角形被细分为四个新的三角形,因此在个细分步骤之后,一个三角形已细分为个三角形.

首先,让我们关注现有的顶点,其中是细分步数.这意味着是控制网格的顶点.

经过一次细分,变为.通常,,其中是极限点.如果具有个相邻顶点,即,那么我们说的**价[valence]**为.有关上述符号,请参见图17.44.同样,价数6的顶点称为规则或普通.否则,它称为不规则或奇点.

下面,给出了Loop方案的细分规则,其中第一个公式是将现有顶点更新为,第二个公式是在与每个顶点之间创建新顶点.同样,是的价:

请注意,我们假设索引是以模为单位计算的,因此,如果,则对于,我们使用索引0;同样,当时,对于,我们使用索引.这些细分规则可以很容易地可视化为蒙版,也称为模板.见图17.45.这些方法的主要用途是仅使用简单的图示就可以传达几乎整个细分方案.请注意,两个蒙版的权重之和为一.这是所有细分方案均适用的特征,其理由是,新点应位于加权点附近.在公式17.61中,常数实际上是的函数,并由下式给出：

Loop对函数的建议[**1067**]给出了每个规则顶点的连续性表面,以及其他地方[**1976**],即所有不规则顶点的连续性表面.由于在细分过程中仅创建了常规顶点,因此在控制网格中存在不规则顶点的位置,曲面仅是.关用Loop方案细分的网格的示例,请参见图17.46.Warren和Weimer [1847]给出了避免三角函数的公式17.62的一种变式:

对于规则价,这将提供曲面,并在其他地方给出.生成的曲面很难与常规Loop曲面区分开.对于未闭合的网格,我们无法使用所提供的细分规则.相反,必须对此类边界使用特殊规则.对于Loop方案,可以使用公式17.60的反射规则.第17.5.3节也对此进行了处理.

经过无数次细分步骤后的曲面称为极限曲面.极限曲面点和极限切线可以使用闭合形式的表达式来计算.使用公式17.61的第一行可以计算顶点的极限位置[**767,1977**],并将替换为

可以通过对紧邻的相邻顶点(称为1环或1邻域)进行加权来计算顶点的两个极限正切,如下所示[767,1067]:

则法线为.注意,这通常比第16.3节中描述的方法要便宜[1977],后者需要计算相邻三角形的法线.更重要的是,这给出了该点的精确法线.

近似细分方案的主要优点在于,生成的曲面趋于公平.从广义上讲,公平与曲线或曲面的弯曲程度有关[**1239**].较高的公平性意味着曲线或表面更平滑.另一个优点是近似方案的收敛速度比插值方案快.但是,这意味着形状经常收缩.这对于较小的凸形网格最为明显,例如图17.47中所示的四面体.减少这种影响的一种方法是在控制网格中使用更多的顶点,即在建模时必须小心. Maillot和Stam提出了一个组合细分方案的框架,以便可以控制缩小[**1106**].有时可以使用的一个优点是,Loop曲面包含在原始控制点的凸包内部[**1976**].

Loop细分方案可生成广义的三向四次方盒样条.因此,对于仅包含规则顶点的网格,我们实际上可以将曲面描述为样条曲面的一种.但是,这种描述对于不规则的设置是不可能的.能够从任何顶点网格生成平滑表面是细分方案的一大优势.有关使用Loop方案的细分曲面的不同扩展,另请参见第17.5.3和17.5.4节.

17.5.2 Catmull-Clark细分 2020年11月5日12点33分

Catmull-Clark[**239**]和Doo-Sabin[**370**]是可以处理多边形网格(而不仅仅是三角形)的两个最著名的细分方案.在这里,我们仅简要介绍前者.皮克斯的短片《杰瑞的游戏》[**347**],《玩具总动员2》以及皮克斯随后所有的故事片都使用了Catmull-Clark表面.这种细分方案也通常用于制作游戏模型,并且可能是最受欢迎的一种.正如DeRose等人[347]指出的那样,Catmull-Clark表面倾向于生成更对称的表面.例如,一个长方形的盒子会产生一个对称的椭圆形表面,与直觉相吻合.相反,基于三角形的细分方案会将每个立方体面视为两个三角形,并因此根据正方形的分割方式生成不同的结果.

Catmull-Clark曲面的基本概念如图17.48所示,Catmull-Clark细分的实际示例如图17.41第757页所示.可以看出,该方案仅生成具有四个顶点的面.实际上,在第一细分步骤之后,仅生成价数4的顶点,因此将这些顶点称为普通或规则的(与三角形方案的价数6相比).

遵循Halstead等人[**655**]的表示法,让我们关注具有个周围边缘点的顶点,其中.参见图17.49.现在,对于每个面,计算新的面部点作为脸部质心,即面部点的均值.鉴于此,细分规则为[**239,655,1977**]

可以看出,顶点被计算为所考虑顶点的权重,边缘点的平均值以及新创建的面点的平均值.另一方面,新的边缘点是通过考虑的顶点,边缘点和以边缘为邻居的两个新创建的面点的平均值来计算的.

Catmull-Clark表面描述了广义的三次B样条曲线表面.因此,对于仅包含规则顶点的网格,我们实际上可以将其表面描述为双三次B样条表面(第17.2.6节)[**1977**].但是,这对于不规则的网格设置是不可能的,并且能够使用细分表面来处理这些设置是该方案的优势之一.即使使用显式公式[1687]在任意参数值下也可以计算极限位置和切线.Halstead等[**655**]描述了一种计算极限点和法线的不同方法.

有关可以使用GPU渲染Catmull-Clark细分曲面的一组有效技术,请参见第17.6.3节.

17.5.3 分段平滑细分 2020年11月5日14点14分

从某种意义上说,曲面可能因为缺少细节而被认为很无聊.改善此类表面的两种方法是使用凹凸贴图或位移贴图(第17.5.4节).这里介绍了第三种方法,即分段平滑细分.基本思想是更改细分规则,以便可以使用飞镖,拐角和折痕.这增加了可以建模和表示的不同表面的范围.Hoppe等人[**767**]首先对Loop的细分曲面进行了描述.请参见图17.50,以比较标准Loop细分曲面和具有分段平滑细分的曲面.

为了能够在表面上实际使用这些特征,首先要标记要锐化的边缘,因此我们知道在何处细分.到达顶点的尖锐边缘的数量表示为.然后将顶点分类为:平滑(),缝纹(),折痕()和角().因此,折痕是表面上的曲线,曲线的连续性为.缝纹是一个无边界的折线m折痕在此结束并平滑地融合到曲面中.最后,角是三个或更多折痕在一起的顶点.可以通过将每个边界边缘标记为尖锐来定义边界.

在对各种顶点类型进行分类之后，Hoppe等人使用一张表格来确定用于各种组合的蒙版.它们还显示了如何计算极限曲面点和极限切线.Biermann等[**142**]提出了几种改进的细分规则.例如,当非凡顶点位于边界上时,先前的规则可能会导致间隙.新规则可以避免这种情况.同样,它们的规则使在顶点处指定法线成为可能,并且生成的曲面将适应于在该点获得该法线.DeRose等[**347**]提出了一种创建软折痕的技术.它们允许先将边缘细分为锐度多次(包括分数),然后再使用标准细分.

17.5.4 位移细分 2020年11月5日14点28分

凹凸贴图(第6.7节)是向平滑表面添加细节的一种方法.但是,这只是一个幻觉,它改变了每个像素的正常或局部遮挡.带有或不带有凹凸贴图的对象的轮廓看起来相同.凹凸贴图的自然扩展是位移贴图[**287**],其中表面被位移.这通常是沿着法线方向完成的.因此,如果曲面的点为,并且其归一化法线为,则位移曲面上的点为

标量d是在点的位移.位移也可以是矢量值[**938**].

在本节中,将介绍位移细分表面[**1006**].总体思路是将位移表面描述为粗略的控制网格,将其细分为平滑表面,然后使用标量场沿其法线进行位移.在位移细分表面的情况下,公式17.67中的是细分表面(粗糙控制网格的边界)上的极限点,是处的归一化法线,计算如下

在公式17.68中,和是细分表面的一阶导数.因此,他们在处描述了两个切线.Lee等人[1006]将Loop细分表面用于粗糙控制网格,可以使用公式17.65计算其切线.请注意,此处的表示法略有不同.我们使用和来代替和.公式17.67描述了所得曲面的位移位置,但我们还需要在位移细分曲面上具有法线才能正确渲染该曲面.解析计算如下所示[1006]：

为了简化计算，Blinn [160]建议,如果位移较小,则可以忽略第三项.否则,以下表达式可用于计算(类似)[1006]:

请注意,不是任何新的表示法,它只是计算中的“临时”变量.对于普通顶点(价),一阶和二阶导数特别简单.它们的遮罩如图17.51所示.对于非寻常顶点(价),省略了公式17.69中第一行和第二行中的第三项.在图17.52中显示了将位移映射与Loop细分一起使用的示例.

当位移的表面远离观察者时,可以使用标准的凹凸贴图来产生这种位移的错觉.这样做可以节省几何处理.一些凹凸贴图方案需要在顶点处有切线空间坐标系,并且可以将以下内容用于()，其中t = pu / || pu || 并且b = n×t。

Nießner和Loop[**1281**]提出的方法与Lee等人提出的方法类似,但是他们使用Catmull-Clark曲面并直接评估位移函数上的导数,这更快.他们还使用硬件细分管道(第3.6节)进行快速细分,

17.5.5 法线,纹理和颜色插值 2020年11月5日14点50分

在本节中,我们将介绍用于处理法线,纹理坐标和每个顶点颜色的不同策略.

如第17.5.1节中的Loop方案所示，极限切线和极限法线可以显式计算.这涉及三角函数,这些函数的评估成本可能很高.Loop和Schaefer[**1070**]提出了一种近似技术,其中Catmull-Clark曲面始终由双三次Béezier曲面近似(第17.2.1节).对于法线,得出两个切线面片,即一个在u方向上,一个在v方向上.然后找到法线作为这些向量之间的叉积.通常,使用公式17.35计算B´ezier面片的导数.但是,由于派生的B´ezier面片近似于Catmull-Clark表面,因此切线面将不会形成连续的法线场.有关如何克服这些问题,请参阅Loop和Schaefer的论文[1070]. Alexa和Boubekeur[29]认为,在每次计算的质量上细分法线可能会更有效,这可以使阴影具有更好的连续性.有关如何细分法线的详细信息,请参阅他们的论文.Ni等人的SIGGRAPH课程[1275]中也可以找到更多近似值.

假设网格中的每个顶点都具有纹理坐标和颜色.为了能够将这些用于细分曲面,我们还必须为每个新生成的顶点创建颜色和纹理坐标.最明显的方法是使用与细分多边形网格相同的细分方案.例如,您可以将颜色视为四维向量(RGBA),然后对其进行细分以为新顶点创建新颜色.这是一种合理的方法,因为颜色将具有连续的导数(假定细分方案至少为C1),因此避免了表面上颜色的突然变化.对于纹理坐标当然可以这样做[**347**].但是,当纹理空间有边界时,必须小心.例如,假设我们有两个面块共享一条边,但沿该边具有不同的纹理坐标.通常,应使用表面规则对几何进行细分,但在这种情况下,应使用边界规则对纹理坐标进行细分.

Piponi和Borshukov[**1419**]提出了一种用于细分表面纹理的复杂方案.

17.6 高效细分 2020年11月5日14点55分

为了在实时渲染上下文中显示曲面,我们通常需要创建该曲面的三角形网格表示.此过程称为细分[tessellation].细分的最简单形式称为均匀细分.假设我们有一个参数B´ezier面片,如公式17.32中所述.我们希望通过在每个面片上计算11个点来细分该面片,从而形成10×10×2 = 200个三角形.最简单的方法是对空间进行均匀采样.因此,我们对所有求，其中和都可以是0到10之间的任何整数.这可以通过两个嵌套的for循环来完成.可以通过四个表面点和创建两个三角形.

尽管这很简单,但是有更快的方法可以做到.与其通过从CPU到GPU的总线发送由许多三角形组成的细分曲面,更有意义的是将曲面表示发送到GPU并让其处理数据扩展.回想一下,细分阶段已在3.6节中进行了描述.有关快速更新,请参见图17.53.

细分器可以使用分数细分技术,这将在以下部分中进行介绍.接下来是关于自适应镶嵌的部分,最后,我们描述如何使用镶嵌硬件渲染Catmull-Clark曲面和位移映射曲面.

17.6.1 局部细分 2020年11月5日14点56分

为了获得更平滑的参数表面细节水平,Moreton引入了分数细分系数[**1240**].由于可以在参数化曲面的不同侧面上使用不同的细分因子,因此这些因素使得自适应细分的形式有限.在这里,将概述这些技术的工作原理.

在图17.54中,左侧显示了行和列的恒定镶嵌因子,而在右侧显示了所有四个边缘的独立镶嵌因子.请注意,边缘的镶嵌因子是在该边缘上生成的点数减去一.在右侧的补丁中,这两个边缘在内部使用的顶部和底部因子越大,类似地,在内部使用的左侧和右侧因子也越大.因此,基本镶嵌速率为4×8.对于系数较小的侧面,沿边缘填充了三角形.Moreton[1240]更详细地描述了这个过程.

细分曲面细分因数的概念如图17.55所示.对于整数细分因子,以生成个点,其中.对于分数细分因子,以生成点,其中.此处,计算的上限,它是朝最接近的整数,而计算底数,它是朝-∞最接近的整数.然后,最右边的点只是“捕捉”到最右边的端点.从图17.55的中间插图可以看出,该模式不对称.这导致问题,因为相邻的贴片可能在另一个方向上生成点,从而在表面之间产生裂纹.Moreton通过创建对称的点图案来解决此问题,如图17.55的底部所示.有关示例,请参见图17.56.

到目前为止,我们已经看到了对具有矩形区域（例如B'ezier面片）的曲面进行细分的方法.但是,也可以使用分数[**1745**]对三角形进行细分,如图17.57所示.像四边形一样,也可以为每个三角形边缘指定独立的分数细分速率.如前所述,这可以实现自适应镶嵌(第17.6.2节),如图17.58所示,其中绘制了位移映射地形.一旦创建了三角形或四边形,就可以将它们转发到管道中的下一步,在下一小节中进行处理.

17.6.2 自适应细分 2020年11月5日15点02分

如果采样率足够高,则均匀镶嵌效果会产生良好的结果.但是,在表面上的某些区域可能不需要其他区域那样的高度镶嵌.这可能是因为表面在某些区域弯曲得更快,因此在那里可能需要更高的细分,而表面的其他部分几乎是平坦的或很远,仅需要几个三角形即可近似它们.生成不必要的三角形问题的一种解决方案是自适应曲面细分,这是指根据某些度量(例如,曲率,三角形边缘长度或某些屏幕尺寸度量)调整曲面细分速率的算法.图17.58显示了适用于地形的自适应镶嵌的示例.

必须注意避免在不同的棋盘格化区域之间出现裂纹.参见图17.59.使用分数细分时,通常将边缘细分因子基于仅来自边缘本身的信息,因为边缘数据在两个连接的面片之间共享.这是一个很好的开始,但是由于浮点数不正确,仍然可能会出现裂缝.Nießner等人[1279]讨论了如何使计算完全不透水,例如,通过确保对于一条边缘,返回完全相同的点,而不管从p0到p1是否进行细分,反之亦然.

在本节中,我们将介绍一些通用技术,这些技术可用于计算分数镶嵌细分率,或决定何时终止进一步的镶嵌细分以及何时将较大的补丁分成一组较小的补丁.

终止自适应镶嵌

为了提供自适应细分,我们需要确定何时停止细分,或者等效地如何计算细分细分因子.您可以仅使用边的信息来确定是否应细分曲面细分,也可以使用整个三角形或组合中的信息.

还应注意的是,如果某个边缘的曲面细分因数从一帧到下一帧变化太大,则使用自适应曲面细分时,帧之间可能会游动或弹出伪影.在计算镶嵌细分因子时,这可能也是要考虑的因素。给定一条具有关联曲线的边,即斑块边缘曲线,我们可以尝试估计曲线在和之间的平坦程度.参见图17.60.找到和之间的参数空间中点,并计算其三维对应点.最终,计算出及其在和之间的直线上的投影之间的长度.该长度用于确定该边缘上的曲线段是否足够平坦.如果足够小,则认为它是平坦的.请注意,此方法可能会错误地认为S形曲线段是平坦的.一种解决方案是随机扰动参数采样点[470].除了使用之外,还可以使用比率,并给出相对度量[404].请注意,该技术也可以扩展为考虑三角形,您可以在其中简单地计算三角形中间的表面点,并使用该点到三角形平面的距离.为了确定这种算法是否终止,通常会设定可以细分多少个上限.当达到该限制时,细分结束.对于小数细分,可以将到的向量投影到屏幕上,并将其(缩放的)长度用作细分率.

到目前为止,我们已经讨论了如何仅从表面形状确定镶嵌率.通常用于动态细分的其他因素包括顶点的局部邻域是否为[769,1935]:

1. 在视图视锥内部.
2. 正面.
3. 在屏幕空间中占据较大的区域.
4. 靠近对象的轮廓.

这些因素将在这里依次讨论.对于视锥平截剔除,可以放置一个球体以包围边缘.然后针对视锥视点测试此球体.如果在外面,我们就不会再细分该边缘.

对于面部剔除,可以从表面描述中计算以及可能的处的法线.这些法线与和一起定义了三个平面.如果全部都是背面的,则该边缘可能不需要进一步细分.

实现屏幕空间覆盖的方法有很多(请参见第19.9.2节).所有方法都将一些简单的对象投射到屏幕上,并估计屏幕空间的长度或面积.较大的面积或较大的长度意味着应进行镶嵌.图17.61显示了从和的线段的屏幕空间投影的快速估计.首先,将线段平移,使其中点位于视线上.然后,假定线段平行于近平面n,并从该线段计算出屏幕空间投影.使用图示右侧的线段和的点,则屏幕空间投影为

分子只是线段的长度.它除以从眼睛到线段中点的距离.然后将计算出的屏幕空间投影与代表屏幕空间最大边缘长度的阈值进行比较.重写前面的公式以避免计算平方根,如果满足以下条件,则细分将继续:

注意,是常数,因此可以预先计算.对于细分曲面细分,可以将公式17.71中的用作曲面细分速率，并可能应用比例因子.测量投影边缘长度的另一种方法是在边缘的中心放置一个球体,使半径为边缘长度的一半，然后将球体的投影用作边缘细分系数[1283].该测试与面积成正比,而上述测试与边缘长度成正比.

增加轮廓的镶嵌速率很重要,因为轮廓对于物体的感知质量起着主要作用.可以通过测试处的法线与从眼睛到的向量之间的点积是否接近零来确定三角形是否在轮廓边附近.如果对于或中的任何一个都是如此,则应进一步细分.

对于位移细分,Nießner和Loop[**1281**]对每个基础网格顶点使用以下因子之一,该顶点与个边向量:

其中循环索引遍历连接到的所有个边,是摄像机的位置,是用户提供的常数.在这里,仅基于距相机的顶点距离,计算与连接的四边形的面积,使用最大边缘长度.然后,将顶点的镶嵌因子计算为边的两个基本顶点的镶嵌因子中的最大值.内部细分因子计算为相对边缘的细分因子的最大值(以和表示).此方法可以与本节中介绍的任何边缘细分因子方法一起使用.

值得注意的是，Nießner等人[**1279**]建议对字符使用单个全局镶嵌因子,该因子取决于与字符的距离.细分数为,其中f是每个字符的细分系数，可以使用上述任何一种方法来计算.

很难说什么方法将在所有应用程序中起作用.最好的建议是测试几种提出的启发式方法,以及它们的组合.

分割和骰子方法

Cook等人[**289**]引入了一种称为分割和切块的方法,其目的是细分表面,使每个三角形变成像素大小,以避免几何混淆.出于实时目的,该细分阈值应增加到GPU可以处理的阈值.首先将每个面片递归地分成一组子面片,直到估计如果对某个子面片使用统一的细分,则三角形将具有所需的大小.因此,这也是自适应镶嵌的一种.

想象有一个大的补丁用于地形.通常,例如,没有办法使小部分细分能够适应,使得相机越接近细分率越高,而细分距离越低.因此,即使目标细分速率的三角形比像素大小的三角形大,分割和切块的核心也可能对实时渲染有用.

接下来，我们描述在实时图形场景中进行拆分和切块的一般方法.假设使用了矩形补丁.然后从整个参数域(即从到的方块)开始递归例程.使用刚刚描述的自适应终止标准,测试表面是否足够细分.如果是,则终止镶嵌.否则,将该域划分为四个不同的相等大的正方形,并针对四个子正方形中的每个递归调用例程.递归继续进行操作,直到曲面被充分细分或达到预定义的递归级别为止.该算法的性质意味着在细分期间会递归创建四叉树.但是,如果相邻的子正方形被细分为不同的级别,则会产生裂纹.标准解决方案是确保两个相邻的子正方形最多只有一个级别不同.这称为受限四叉树.然后,使用图17.59右侧所示的技术填充裂缝.这种方法的缺点是簿记更多.

Liktor等人[**1044**]提出了GPU分割和切块的变体.问题在于,当由于例如照相机已移近表面而突然决定再分开一次时,要避免游泳伪影和爆裂效果.为了解决这个问题,他们使用了分数分割方法,并受到了分数细分的启发.如图17.62所示.由于分裂是从一侧朝向曲线中心或朝向贴片侧中心顺利引入的，因此避免了游动和爆裂伪影.当达到自适应细分的终止标准时，GPU还将使用分数细分对每个剩余的子补丁进行细分.

17.6.3 快速Catmull-Clark细分 2020年11月5日15点43分

在建模软件和故事片渲染中经常使用Catmull-Clark表面(第17.5.2节),因此能够使用图形硬件有效地渲染这些表面也很有吸引力.近年来,Catmull-Clark表面的快速镶嵌方法一直是活跃的研究领域.在这里,我们将介绍一些这些方法.

近似方法

Loop和Schaefer[**1070**]提出了一种将Catmull-Clark曲面转换为可在域着色器中快速求值的表示形式的技术,而无需了解多边形的邻居.

如第17.5.2节所述,当所有顶点均为普通顶点时,Catmull-Clark曲面可描述为许多小的B样条曲面.Loop和Schaefer将原始Catmull-Clark细分网格中的四边形(四边形)多边形转换为双三次B´ezier曲面（第17.2.1节）.对于非四边形来说这是不可能的,因此我们假定不存在这样的多边形(回想一下,在细分的第一步之后,只有四边形多边形）。当顶点的化合价不同于4时，就不可能创建与Catmull-Clark表面相同的双三次B´ezier面片。因此，提出了一种近似表示形式，该表示形式对于具有价-四个顶点的四边形非常精确，并且在其他位置靠近Catmull-Clark表面。为此，将同时使用几何面片和切线面片。

几何面片只是具有4×4个控制点的双三次B´ezier面片.我们将描述如何计算这些控制点. 完成此操作后,可以对补丁进行细分,并且域着色器可以在任何参数坐标（u，v）上快速评估B´ezier补丁。 因此，假设我们有一个仅包含具有四个价顶点的四边形的网格，我们要针对该网格中的某个四边形计算对应的B´ezier面片的控制点。 为此，需要四边形附近。 执行此操作的标准方法如图17.63所示，其中显示了三个不同的蒙版。 可以旋转并反映这些内容，以创建所有16个控制点。 注意，在一个实施方式中，掩模的权重应该加起来为一，为清楚起见，这里省略了该过程。

在通常情况下，以上技术会计算出Béezier补丁。 当至少有一个非凡顶点时，我们计算一个非凡面片[1070]。 遮罩如图17.64所示，其中灰色四边形的左下顶点是非凡顶点。

请注意，这会产生一个贴近Catmull-Clark细分曲面的面片，并且沿具有非凡顶点的边仅C0。 添加阴影时，这通常看起来会分散注意力，因此建议使用与N补丁（第17.2.4节）类似的技巧。 但是，为降低计算复杂度，导出了两个切线补丁：一个在u方向上，一个在v方向上。 然后找到法线作为这些向量之间的叉积。 通常，使用公式17.35计算B´ezier面片的导数。 但是，由于派生的B´ezier面片近似于Catmull-Clark表面，因此切线面将不会形成连续的法线场。 有关如何克服这些问题，请参阅Loop和Schaefer的论文[1070]。 图17.65显示了可能发生的伪像类型的示例。

Kovacs等人[931]描述了如何扩展上述方法以处理折痕和拐角（第17.5.3节），并在Valve的Source引擎中实现这些扩展。

功能自适应细分和OpenSubdiv

皮克斯提出了一个名为OpenSubdiv的开源系统，该系统实现了一组称为特征自适应细分（FAS）的技术[1279、1280、1282]。基本方法与刚才讨论的先前技术有很大不同。这项工作的基础在于，细分相当于规则面的双三次B样条斑块（第17.2.6节），即每个顶点都是规则的四边形，这意味着该顶点的化合价为4。因此，细分仅针对非规则面继续进行递归，直到达到某个最大细分级别为止。如图17.66左侧所示。 FAS还可以处理折痕和半平滑折痕[347]，并且FAS算法也需要围绕此类折痕进行细分，如图17.66右侧所示。可以使用曲面细分管线直接绘制双三次B样条曲线补丁。

该方法首先使用CPU创建一个表。该表将索引编码到细分期间直至指定级别需要访问的顶点。这样，由于索引独立于顶点位置，因此可以对基础网格进行动画处理。一旦生成了三次B样条曲线补丁，就无需继续递归，这意味着表格通常会变得相对较小。基本网格和带有索引以及其他化合价和折痕数据的表一次上传到GPU。

要一步细分网格，首先计算新的面点，然后计算新的边缘点，最后更新顶点，并对每种类型使用一个计算着色器。为了进行渲染，在完整补丁和过渡补丁之间进行了区分。完整面片（FP）仅与相同细分级别的面片共享边，而常规FP使用GPU细分管线直接渲染为双三次B样条面片。否则继续细分。自适应细分过程可确保相邻小块之间最多只有一个细分级别差异。过渡补丁（TP）与至少一个邻居的细分级别有所不同。为了获得无裂纹的渲染，每个TP都分成几个子补丁，如图17.67所示。这样，棋盘格化的顶点沿着每个边的两侧匹配。每种类型的子修补程序均使用实现插值变体的不同外壳和域着色器进行渲染。例如，图17.67中最左边的情况被渲染为三个三角形B样条斑块。在非正常顶点周围，使用了另一个域着色器，其中使用Halstead等人的方法计算极限位置和极限法线。 [655]。使用OpenSubdiv渲染Catmull-Clark曲面的示例如图17.68所示。

FAS算法可处理折痕，半平滑折痕，分层细节和自适应细节级别。 有关更多详细信息，请参阅FAS论文[1279]和Nießner的博士学位论文[1282]。 Schâafer等人[1547]提出了一种称为DFAS的FAS变体，它的速度更快。

自适应四叉树

Brainerd等[190]提出了一种称为自适应四叉树的方法。它与Loop和Schaefer [1070]的近似方案相似，在原始基础网格的每个四边形中都提交了一个细分的图元。此外，它预先计算细分计划，该计划是一个四叉树，它对从输入面到某个最大细分级别的分层细分（类似于特征自适应细分）进行编码。细分计划还包含细分面所需的控制点的模板掩码列表。

在渲染期间，遍历四叉树，从而可以将（u，v）坐标映射到细分层次结构中的补丁，可以直接对其进行评估。四叉树叶是原始面域的子区域，可以使用模具中的控制点直接评估该子区域中的表面。迭代循环用于遍历域着色器中的四叉树，其输入是参数（u，v）坐标。遍历需要继续，直到到达（u，v）坐标所在的叶节点为止。根据四叉树中到达的节点类型，将采取不同的操作。例如，当到达可以直接评估的子区域时，将为其对应的双三次B样条曲线补丁获取16个控制点，并且着色器将继续评估该补丁。

有关使用此技术渲染的示例，请参见第718页的图17.1。 此方法是迄今为止最快的方法，它可以精确地渲染Catmull-Clark细分曲面，并处理折痕和其他拓扑特征。 在FAS上使用自适应四叉树的另一个优势如图17.69所示，并在图17.70中进一步显示。 自适应四叉树还提供更统一的镶嵌，这是因为在每个提交的四边形和镶嵌图元之间存在一对一的映射。

更多阅读和资源

曲线和曲面的主题非常广泛，有关更多信息，最好查阅仅关注此主题的书籍。 Mortenson的书[1242]很好地介绍了几何建模。 Farin [458，460]以及Hoschek和Lasser [777]所著的书都是通用的，涉及计算机辅助几何设计（CAGD）的许多方面。对于隐式表面，请参阅Gomes等人的书。 [558]和de Ara´ujo等人的最新文章[67]。有关细分表面的更多信息，请查阅Warren和Heimer的书[1847]，以及Zorin等人[1977]关于“建模和动画细分”的SIGGRAPH课程注释。 Ni等人[1275]的细分表面替代课程在这里也是有用的资源。 Nießner等人[1283]和Nießner的博士学位论文[1282]进行的这项调查对于使用GPU实时绘制细分曲面的信息非常有用。对于样条插值，除了上面的Farin [458]和Hoschek and Lasser [777]的书以外，我们还请感兴趣的读者阅读Killer B的书[111]。高盛[554]给出了曲线和曲面的伯恩斯坦多项式的许多性质。 Farin的文章[457]几乎包含了您需要了解的有关三角形Béezier曲面的所有信息。另一类有理曲线和曲面是非均匀有理B样条（NURBS）[459、1416、1506]，通常在CAD中使用。