

建筑结构建模系统的研制

林小禾

(浙江科技学院 机械与汽车工程学院,杭州 310023)

摘 要: 介绍了利用 VC++ 研制可视化空间结构系统分析程序 VSAP 的某些关键技术,阐述了 VSAP 程序的主要功能,简要地叙述了 VSAP 程序的主要技术特点及各部分的构成,讨论了编制 CAD 程序经常用到的内存管理和数据存储方法,并给出了程序代码。此外,还讨论了在建模过程中难以处理的椭圆和椭圆弧的构造问题,最后给出一个例子证明 VSAP 程序建模是有效的。

关键词: 建筑结构;结构分析;前处理;建模;内存管理;数据存储

中图分类号: TU311.41;TP311.11

文献标识码: A

文章编号: 1671-8798(2006)01-0039-06

Study on the Modeling System of Building Structure

LIN Xiao-he

(School of Mechanical and Automotive Engineering, Zhejiang University of Science and Technology, Hangzhou 310023, China)

Abstract: This paper introduced some key technique in the visual space structure analysis program system with VC++. The main function of the VSAP program system was introduced. The main technique characteristics and each part of composings of the VSAP program system was briefly described. It discussed the method of the memory management and data storage usually used in the CAD programmer. Furthermore, it also discussed the problem of establishing ellipse and ellipse arc which is very difficulty in the course of modeling. An example was given to prove the availability at last.

Key words: building structure; structure analysis; pre-treat; modeling; memory management; data storage

随着社会经济的发展和人们物质生活水平的提高,现代建筑向复杂化、大型化发展。如何保证计算模型的合理性及计算结果的可靠性,已成为结构工程师面对的首要问题。

在实际计算中,建筑结构中相当大一部分的构件均简化为杆件(例如框架结构中的梁、柱以及空间

网格结构中的杆等)。在结构设计中,必须对它们进行相应的计算以确定所设计的结构是否安全可靠。早先,计算程序需要的输入数据文件是由用户编写后再输入到计算机中去,对于比较复杂的结构,这种过程是极其繁琐且容易出错的。现代的建筑结构越来越复杂,规模也越来越大,一般的建筑结构往往有

收稿日期: 2005-12-09

基金项目: 浙江省科技计划项目(2006C31057);浙江省教育厅科研计划项目(20041171)

作者简介: 林小禾(1952—),男,福建闽侯人,讲师,硕士,主要从事力学教学工作和建筑结构 CAD 软件研制。

成千上万根杆件,在这种情况下,再用手工编写输入数据文件是不现实的。因此,有必要研制具有可视化的图形处理和功能强大的数据处理能力的计算机程序,利用这种程序建立空间结构模型,然后由它形成计算程序所需要的输入数据文件,这就是结构分析的前处理部分,它是现代结构分析的重要组成部分之一。本文主要阐述用 VC++ 编制结构分析程序的前处理部分。

1 结构分析程序前处理功能的实现

当前,所有的结构分析程序都有前处理部分,但许多程序的前处理功能不强,主要体现在建模过程麻烦,自动化程度不高,而且修改不方便。此外,还缺少空间网壳和框架共存结构的设计计算程序。

由于教学、科研工作以及某些较为特殊建筑结构的设计和计算的需要,研制开发了 VSAP 系统。该系统带有自己的前后处理和计算分析模块,可以应用在高校的教学和科研方面,也可以应用在建筑设计单位的结构设计和计算方面。

1.1 前处理程序的设计思路

VSAP 前处理程序属于图形应用程序,在设计时,主要从以下几个方面进行考虑和设计。

1.1.1 定义基类和基本图元类 首先,定义一个从 CObject 类派生的实体图形的抽象基类 CEntity,它具有各种图形的基本属性,如颜色、线型和线宽等,并有一些设置和获取操作,在基类中还定义了图形绘制、编辑和修改(如求交、剪切、延伸、镜像、阵列、拷贝等)时需要用到的函数,这些函数都被定义为虚函数。然后定义基类 CEntity 的派生类——直线类(CLine)、圆类(CCircle)、圆弧类(CArc)、椭圆类(CEllipse)和椭圆弧类(CEllipseA)。对于杆件则分别定义节点类(CNode)和杆件单元类(CElem)。在定义基类和派生类时,对于所有派生类共有的公共功能,放在基类中,这样所有的派生类都可以继承它,而对于某类特有的功能,则把它加入到该派生类中。在基类中将图形绘制、编辑和修改的函数定义为虚函数,由于基类的指针可以指向派生类的指针,当用指向基类的指针对虚函数进行访问时,系统将根据基类指针所指的对象调用该对象的相应函数。此外,在定义节点类和杆件单元类时,把节点载荷、节点约束信息、节点铰链信息和单元载荷等数据均作为相应类的成员变量。这样做的优点是在编辑修改结构时,若某些节点或杆件单元被删除,则上述数据也随之被删除,不会引起混乱。

1.1.2 内存管理、数据存储和文件管理 在程序中,有时需要用到一些临时数组,如果这些临时数组的大小事先可以确定,则可在相应函数中直接定义它们,这种数组当相应函数使用结束后会自动删除,不会造成内存泄露。但这种数组由于是在“栈空间”分配的,而由于“栈空间”的有限性,较大的数组不能使用,另外对于预先不知道具体大小的数组在使用上也不方便。

对于预先不知道具体大小的数组,可采用“堆内存”分配的方法,对于一维情况的“堆内存”分配,所有 C++ 教材上均有介绍,这里不赘述。这里主要介绍一下二维数组的分配。例如在形成框架模型时,如果根据用户的输入数据统计出的杆件数和节点数分别为 gjs 和 jds,则采用如下的方法分配临时数组:

```
//分配单元定义数组
int ** elem=new int * [gjs];
for(kk=0;kk<gjs;kk++) elem[kk]=new int[2];
//分配节点坐标数组
double ** coor=new double * [jds];
for(kk=0;kk<jds;kk++) coor[kk]=new double[3];
```

如果要将这种数组传到另外一个函数中,则可采用如下方法:首先在对应头文件中定义要被调用的函数

```
void function1 (int ** elem);
然后在定义了临时数组的函数中调用
void function0 ()
{
    .....
    function1 (elem);
}
```

这种数组在使用结束后一定要及时删除,否则会造成内存泄露。

```
//释放分配的空间
for(kk=0;kk<gjs;kk++) delete elem[kk];delete elem;
for(kk=0;kk<jds;kk++) delete coor[kk];delete coor;
```

对于建模时得到的实际结构数据,将它们保存在链表中。为此,在文档类中建立一个 CObList 类型的链表^[1],它比 C++ 数组更加健壮灵活,而且节点数、杆件数和图元数可以不受任何限制。

```
// VSAPDoc.h : interface of the CVSAPDoc class
.....
class CVSAPDoc : public CDocument
{
    .....
public:
```



```
COBList    m_EntityList; // 图元对象链表
```

```
};
```

数据存储和文件管理中,采用了二进制的序列化和反序列化的方法。由于实体图形类是从 COBject 类派生的,因此,只要在每个类声明中包含宏 DECLARE_SERIAL,并在相应类的实现文件中包含宏 IMPLEMENT_SERIAL,然后再添加相应的序列化函数,就可以实现序列化和反序列化了。

1.1.3 基本图元的编辑、修改功能 在结构模型的创建、编辑和修改过程中的许多操作都归结为基本图元的求交点运算。这些算法在许多计算机图形学书籍中都可以找到^[2],但是涉及到椭圆和椭圆弧的算法比较复杂,而且 VC++ 的图形函数中提供的椭圆和椭圆弧,其长短轴必须是水平和铅垂的,如果希望可以构造长短轴沿任意方向的椭圆,并简化椭圆(弧)与其他图元之间的求交运算,可以以折线来逼近椭圆(弧),以椭圆为例,首先在相应头文件中定义椭圆折线类:

```
class CEllipse : public CEntity
{
//椭圆折线类
    DECLARE_SERIAL(CEllipse)
public:
    Position    m_center; // 椭圆中心
    double Axes1, Axes2, Ang; // 椭圆的长轴、短轴长度和长轴与 x 方向角度
public:
    CEllipse ();
    //由中心,长轴,短轴和长轴与 x 方向角度形成椭圆
    CEllipse (const Position& center, const double& L1, const double& L2, const double& Alp);
    ~ CEllipse ();
    double bjdd[122][2]; // 折线上边界顶点的 x, y 坐标
    .....
}
```

其中,Position 是预先定义的位置类,用于存储点的坐标(x, y, z),其定义如下:

```
//定义位置类
class Position {
public:
    double x;
    double y;
    double z;
    .....
}
```

然后在椭圆折线类的实现函数中加入如下构造函数:

```
//由中心,长轴,短轴和长轴与 x 方向角度形成椭圆
CEllipse::CEllipse (const Position& center, const double& L1, const double& L2, const double& Alp)
{
    int ii;
    double xx, yy, alp=3.0 * PI/180. ; //PI=acos(-1),即圆周率 π
    Init(); //初始化函数
    m_center = center ; Ang=Alp; Axes1=L1; Axes2=L2;
    for(ii=0; ii<120; ii++)
    {
        xx=L1 * cos((ii+1) * alp); yy=L2 * sin((ii+1) * alp);
        bjdd[ii][0]=m_center.x+xx*cos(Ang)-yy*sin(Ang);
        bjdd[ii][1]=m_center.y+xx*sin(Ang)+yy*cos(Ang);
    }
    bjdd[72][0]=bjdd[0][0]; bjdd[72][1]=bjdd[0][1];
}
```

最后将各边界顶点以简单的直线相连接,这样就构造了一个由 120 段线段组成的封闭折线段,于是椭圆与其他图元的相交运算就可以化为折线段与相应图元的相交运算,这大大降低了求交点的难度。由实际建模过程可知,无论从图形显示还是形成的数据来看,近似程度都是符合工程要求的。

1.2 逐层形成空间框架

VSAP 程序可以分析空间网壳和空间框架以及它们共存结构的问题,限于篇幅这里只介绍空间框架的形成过程。

大部分框架结构是由许多层组成的,各层之间以铅直的柱相连。因此,一个实际的空间框架往往在某一层上有比较复杂的几何形状,而在某两层之间则只是一些铅直的柱。在空间框架建模时,可以采用逐层形成的方法。这种方法的基本思路是:首先确定该层的层高,然后在该楼层所在的 xy 坐标平面上绘制由各种图元组合成的平面图形。平面图形经过编辑、修改以后,得到实际框架结构在这一层上的轴线图,该层所有的图形全部绘制结束后,求出所有图元的交点;然后根据某图元在二交点内的部分形成梁单元(如果在两交点之间为直线图元,则生成直梁;如果是曲线则生成曲梁单元),梁的两个端点的 x、y 坐标即为交点的 x、y 坐标,而 z 坐标即为该层的高度。

1.2.1 创建、编辑轴线 一个实际的建筑结构在某

一层上可能非常复杂,必须能够迅速、准确地形成某一层楼面上的结构模型。为达到该目的,程序提供了强大的图元生成、修改和编辑功能,在 VSAP 程序中,基本图元的绘制功能包括直线、圆、圆弧、椭圆和椭圆弧的绘制(每一种基本图元均可以在命令行输入相关数据绘图或用鼠标直接在屏幕上绘图)。为提高绘图速度,提供了绘制连续直线、垂直直线、平行直线、指定角度的直线、旋转直线、同心圆、同心圆弧等功能。为了避免不必要的麻烦,在将新创建的图元存入数据链表前,还要检查该图元是否与已经存在的图元重叠(或部分重叠),如果有重叠现象则提示出错信息,并取消该图元的创建工作。另外,在图元编辑方面提供上述所有图元的修剪、延伸、偏移、删除、拷贝、平行拷贝、移动、镜像、阵列、旋转等功能,并求出各图元的交点(网格点),且可以对网格点进行编辑(如删除网格点,删除网格线和进行网格点移动等)。在选择了逐层形成空间框架命令之后,将弹出如图 1 所示的工具条,用鼠标单击相应的按钮就可以进行相关的操作。

1.2.2 形成一层上的梁单元 当某一层上的所有图元全部绘制和编辑完后,就可以形成该层上的梁单元了。前面已经提到,某段梁的两个端点是该段梁所在的图元与其他图元的交点,在定义相应图元的类时,已经定义了交点(即网格点)数组,以直线类为例:

```
class CLine : public CEntity
{
    DECLARE_SERIAL(CLine)
    .....
public:
    .....
    //每条直线上网格点数目和图元编号
    int Num, Fnum, nn;
    double xjd[121][2]; //直线上网格点的 x, y 坐标
    .....
};
```

这样,求出的交点将存储在具体的每一个图元对象中;此外,求出每一个图元对象的交点之后,还必须对每一个图元对象的交点(网格点)排序。这样才可以按照每一个图元对象上的网格点依次按顺序形成梁单元。在程序内部,各图元的排序按如下约定:对于直线,若二端点的 x 坐标不同,则按 x 坐标从小到大排序,否则按 y 坐标从小到大排序;对圆、圆弧、椭圆和椭圆弧,则按从 x 正向量起的圆心角大小排序。同时,按照杆件系统有限元的要求,两个同平

创建、编辑轴线	
连续直线	绘制垂线
平行直线	定角直线
二线连接网格	
旋转直线	定向定距
求二圆(弧)相切线	
绘制圆	绘椭圆
同心圆	同心圆弧
绘圆弧	椭圆弧
平行/垂线相切弧	
二圆(弧)相切圆弧	
形成网格点	
添附加点	删附加点
删除轴线	轴线偏移
轴线修剪	轴线延伸
删网格点	删网格线
轴线拷贝	平行拷贝
网点移动	属性修改
轴线旋转	轴线镜像
轴线阵列	轴线移动
显示网格点间距	
显示网格点坐标	
形成柱线	
删除柱线	
拷贝前层梁柱	
本层绘制结束	
全部绘制结束	

图 1 创建、编辑轴线

面上的杆件单元不能重叠(或部分重叠);一个杆件的端点不能在另一杆件的中间;因此,每生成一杆件单元,在存入数据链表前,必须检查该杆件与已经存在的杆件之间是否满足上述要求,若不满足则不生成该杆件。图 2 为形成一层上的梁单元时的程序框图。

1.2.3 形成柱单元 形成同一层上的柱单元相对于形成梁单元要简单得多,因为一般框架结构在两层之间是用铅直的柱单元连接的。在图 1 的工具条按钮中选择“形成柱线”命令,即可选取当前层上的节点。在当前层上的节点处按层高布置柱线,布置了柱线的节点和没有布置柱线的节点将以不同的颜色显示以示区别。

由于柱单元只能在当前层的节点上布置,而在形成一层上的梁单元时,单元的端点(即节点)是根

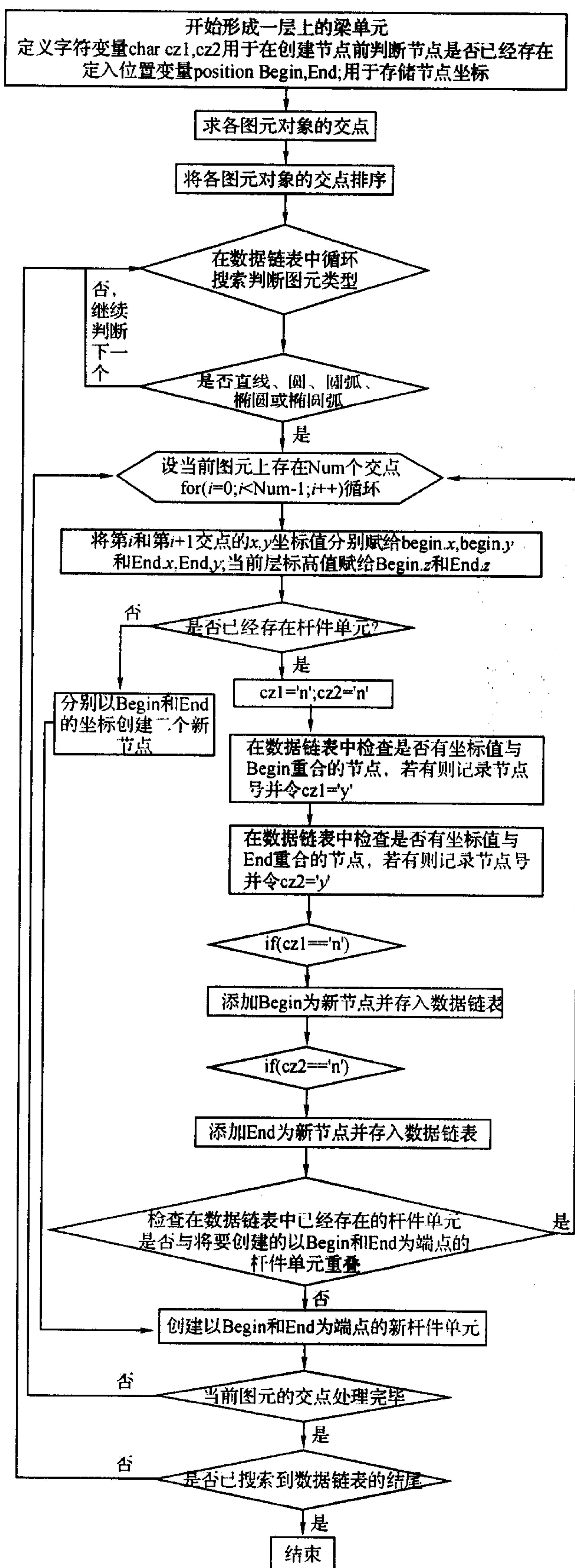


图2 形成一层上梁单元时的程序框图

据形成前各图元的交点(网格点)生成的;若希望在某不是交点的位置布置柱单元,必须先在该处添加一附加网格点,为此程序中实现了这个功能(图1中的“添附加点”命令)。

在形成柱单元前,同样需要检查将要形成的单元与已经存在的单元是否有重叠。

1.2.4 形成后一层梁单元 在建筑结构中,后一层的梁、柱往往与前一层的梁、柱在尺寸与形状上有许多类似之处,它们端点的 x 、 y 坐标相同;而 z 坐标是前一层的 z 坐标加上当前层的层高。在VSAP程序中为充分利用前一层的结果,首先将前一层的所有图元拷贝到当前层,然后在这个基础上进行修改,修改后就可以按照前述相同的方法形成当前层的梁、柱单元了。

2 其他前处理功能

对于框架结构,提供了三种建模方式供用户选用,用户可以根据实际的结构形式选择合适的建模方式。除了上面介绍的“逐层形成空间框架”外,还提供了如下的两种方法。

2.1 直接生成正交形式的三维框架模型

对于许多相对比较规则的正交形式空间框架,用这种方法可以在很短的时间内(数秒钟)生成空间结构的整体轮廓。只要按照程序的提示输入空间框架的横向跨数、纵向跨数和层数,并输入各跨跨长和各层层高,程序就可以立即形成整个空间结构的立体构架,同时,在屏幕上显示出相应的立体结构图形。在形成的整体构架的基础上,还可以方便地进行修改(例如增加、删除某些杆件单元,修改某跨的跨长和某层的层高,添加支座约束和各种载荷等)。

这种方法的程序实现相对比较简单,只需要先根据输入数据计算出框架的结点总数和单元总数,然后按1.1.2的方法分配相应的临时数组,再计算出每层节点数、每层横向梁数、每层纵向梁数和每层杆件数,根据这些数据就可以计算各层的节点和杆件单元数据。计算出的数据先存储在临时数组中,当全部数据计算结束后,再根据这些数据在屏幕上绘制相应的图形,并将它们存入数据链表,最后再按1.1.2的方法删除临时数组。

2.2 形成空间网格

空间网格结构在近年来得到了迅猛发展。由于它可以充分发挥空间三维捷径传力的优越性,特别适宜于覆盖大跨度建筑^[3]。若结构的外形为平板

状,称为网架;其外形为曲面状则称为网壳。

空间网格结构形式有很多,有些大型结构的空间杆件数量可能达到上万根,不可能一根一根地输入杆件的信息。而且由于结构形式的复杂性,采用普通的图形绘制、编辑和修改(如剪切、延伸、镜像、阵列、拷贝等)功能也难以完成它们的建模工作。VSAP 系统提供了许多网格自动生成功能,可以满足大多数工程实际的要求。由于篇幅所限,这里就不详细介绍它们的实现方法了。

3 形成计算程序的输入数据文件

当空间结构建模完成并输入了所有必要的数

以后,就可以形成计算程序所需要的输入数据文件了。在选择了形成输入数据文件命令后,程序从图元对象链表中提取有关数据,按照计算程序所需的格式形成输入数据文件。

4 应用实例

VSAP 系统已经过大量的考题证明了它的正确性,图 3 是用 VSAP 系统建立的一个空间结构模型,该结构的上半部为一个四角锥网壳,下半部为椭圆形框架结构,图中还显示出框架结构上作用的恒载,借助于 VSAP 程序提供的辅助工具,整个结构的建模过程只需要 1 分钟多。

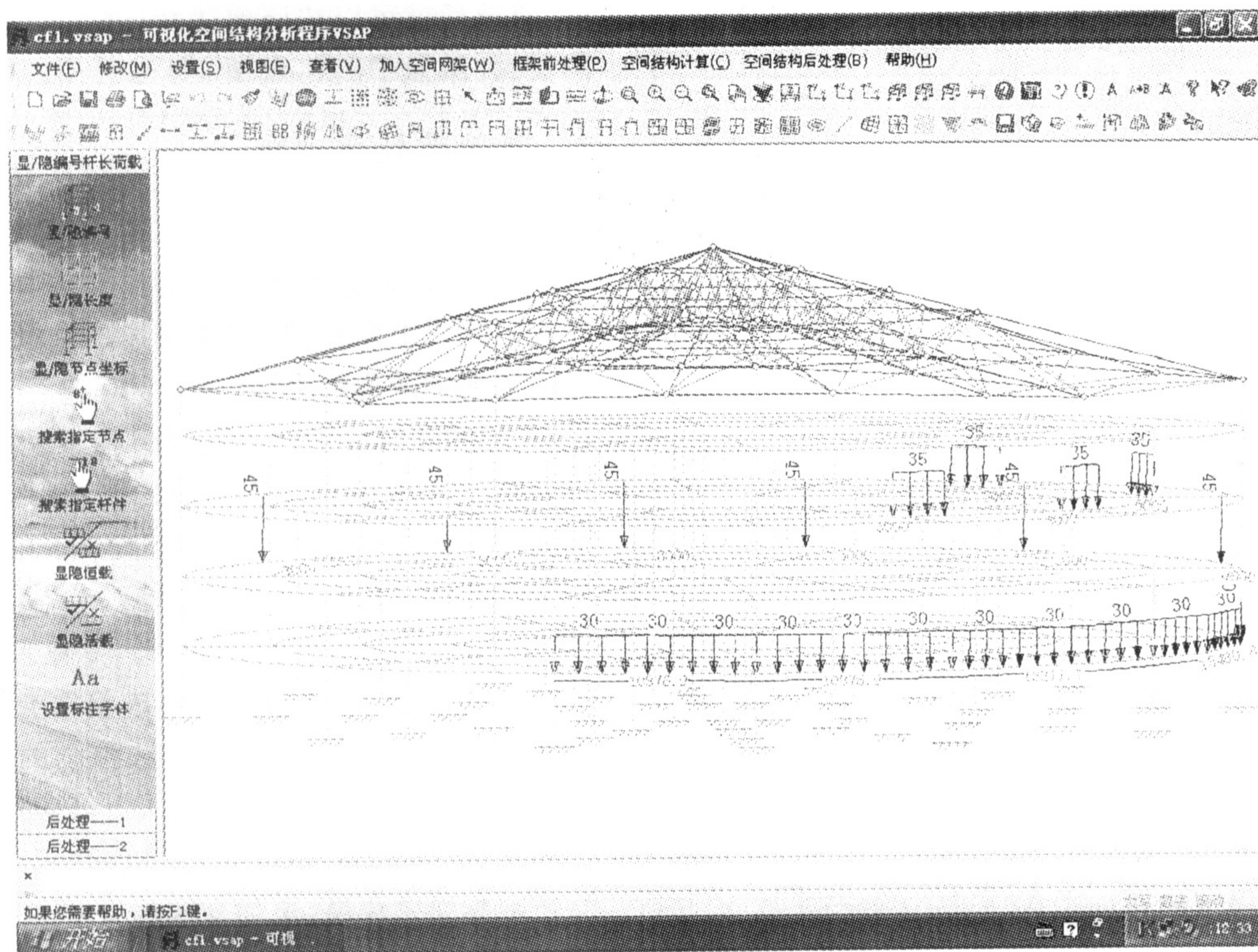


图 3 某空间结构的建模实例

5 结 论

根据对实际空间结构的建模过程,证明用 VSAP 建立空间结构的模型是非常有效的。在建模过程中不但使用方便,速度快,而且修改方便,建模结束后可以立即形成计算模块所需要的输入数据文件。VSAP 系统的开发成功为今后进一步开发更加完善的系统(例如三维实体单元、板壳单元与空间杆系单元混合建模和协同计算)打下了基础。

参考文献:

- [1] 李于剑. Visual C++ 实践与提高——图形图像编程篇[M]. 北京:中国铁道出版社,2001.
- [2] Philip J Schneider, David H Eberly. 计算机图形学几何工具算法详解[M]. 周长发,译. 北京:电子工业出版社,2005.
- [3] 沈世钊. 中国空间结构理论研究 20 年进展[C]//中国土木工程学会桥梁及结构工程分会空间结构委员会. 第十届空间结构学术会议论文集. 北京:中国建材出版社,2002:38-52.