文章编号:1671-4598(2016)07-0256-06 DOI:10.16526/j.cnki.11—4762/tp.2016.07.070 中图分类号:TP391 文献标识码:A

# 车削轴类零件的轮廓轨迹提取与真实感显示

陈智渊
(青海师范大学民族师范学院,青海 海南藏族自治州 813000)

**摘要**:图形数据信息的可视化一直是计算机图形学与可视化领域研究的重点内容,图形数据的挖掘、提取与显示都需要借助计算机图形学的知识来完成;本研究重点探讨了计算机图形真实感处理技术在计算机辅助数控车削自动编程加工系统中的应用,通过运用计算机图形处理技术编程提取DXF轴类零件的轮廓轨迹曲线,采用曲面构造算法作数据处理,生成零件真实感所需的三维数据模型,结合OpenGL内容对三维数据模型进行实体渲染,交互实现了数据模型的光照、材质、雾化等功能,通过旋转、透视变换,零件设计者可真实地观测到零件内部结构,从整体上对所设计的零件结构进行直观的认识;实验结果表明所开发的图形真实感程序系统能够高效的实现车削轴类件的渲染效果,具有较高的轮廓数据提取效率和真实感渲染能力。

**关键词**:可视化;轴类零件;数据处理;真实感;OpenGL

# Trajectory Curve Extraction and Realistic Display of Lathe Turning Shaft Parts

Chen Zhiyuan
(Minorities Teachers College, Qinghai Teachers University, Hainan Tibetan Autonomous Prefecture 813000, China)

**Abstract**: Graphic data information visualization has been a research focus in computer graphics (CG) and visualization field, and graphical data mining, extraction and display are always done with computer graphics knowledge. This study probes into the computer graphics realistic processing technology in computer aided numerical control (NC) lathe turning processing automatic programming system, the application of programming by using computer image processing technology to extract the DXF trajectory curve, the outline of shaft parts and surface construction algorithm is adopted to improve the data processing, generating parts needed for the realistic 3D data model. Combined with OpenGL content for entity rendering of 3D data model, which interactively realized functions of light, material, spray, et al. Through the rotating, perspective transformation, part designers can truthfully observe parts' internal structure, with an intuitive understanding of the whole structure of the designed parts. The experiment results show that the development of graphics realistic application system can efficiently realize lathe turning rendering of shaft parts, and it has the high profile data extraction efficiency and realistic rendering capabilities.

**Keywords**: visualization; shaft part; data processing; realistic; OpenGL

## 0 引言

随着交互式计算机图形学的不断发展,图形图像处理技术已广泛应用于计算机辅助设计、机械加工动画和仿真、科学计算等诸多领域并发挥重要作用[1-3]。通常,计算机图形图像处理技术是基于CAD/CAM等软件,通过运用计算机图形学知识,在计算机上完成图形图像的设计、修改与数据存储,并将数据信息转化为图形或图像在屏幕上进行交互处理与显示的一门综合性技术[4]。现在已经开发了许多硬件设备和数据处理算法来改善生成图形的效率、真实感和速度。当前,计算机图形学的趋势是将更多的物理原理融合到三维图形算法中,更好地模拟物体和真实环境之间的复杂交互[5-6]。

在机械零件的实际加工中,轴类零件一直是车削和镗削常用的加工件,计算机辅助车削、镗削加工系统的主要工作也都是围绕轴类零件进行的。通常,文件图形是依据自定义的专有格式完成数据存储的,整个存储过程并没有一个通用的数据格式规范。各个软件自定义数据结构和文件的储存格式,如UG的PRT格式、CATIA的CGR、IGS和STP格式、Pro/E的PRT格式等。虽然这些文件之间可以通过IGS和STP格式进行数据的相互转换,以方便数据读取与图形显示,但总是在数据导出过程中出现数据量过载或者数据元素大量丢失等问题[7]。相比之下,正是由于DXF文件数据结构更为规范,数据读取更加容易。为此,选取DXF文件作为数据源,就可方便地对图形进行多种真实感效果处理。而要实现轴类件的真实感显示,就需要构造轴类零件。通过分析轴类件的特点可知,轴类件通常为对称模型,且三维零件模型可由1/2的二维轮廓曲线绕轴线旋转获得。这样,构造轴类零件的实质就是获得轴线和母线来构造旋转曲面的过程。本研究就是通过程序读取车削轴类DXF格式文件获得零件的母线(即轮廓曲线)和轴线数据信息,利用曲面构造算法生成三维数据模型,并通过OpenGL图形处理技术对车削轴类件进行真实感显示的实际应用。其中,母线是一条连续的曲线,通过AutoCAD提供的多义线(Polyline)描述,而轴线的识别则是通过设置线型实现的。

## 1 DXF文件数据结构概析

DXF文件包含5段内容信息,分别为标题段、表段、块段、实体段和结束段[8]。这5段内容分别对应存放标题变量、

收稿日期:2016-01-16; 修回日期:2016-02-29。

作者简介:陈智渊(1981-),男,青海乐都人,硕士,中级讲师,主要从事计算机语音图像处理方向的研究。

表信息、块定义实体信息和实体段几何及非几何信息和结束标示。本质上，DXF 文件由众多“组码”和“组值”构成的“数据对”组成。这里的数据对就是通常所说的“组”。每组占两行，组代码在前，作为引行，表示数据类型的名称；组值在后，作为实际内容行，代表着具体的数据信息。两者结合才能完整的表达一个数据的全部信息。在实际应用中，由于 DXF 文件中的实体段包含零件图形全部的几何和插入块信息，是数据的主要来源[9]。因此，对 DXF 图形文件的处理与应用研究，主要深入分析实体段的图元数据信息即可。

实体段的一般格式如下：

0
SECTION
2
ENTITIES
.(单独几何实体信息)
0
ENDSEC

其中几何实体主要包括：POINT、CIRCLE、ARC、TRACE、SOLID、VERTEX、SEQUEND、TEXT、SHAPE、INSERT、ATTDFF、ATTRIB、VIEWPORT 和 3DFACE 以及。由于多义线（POLYLINE）和直线（LINE）中分别存放着母线和轴线的关键信息，是程序读取的关键环节，需要进一步探讨分析。直线与多义线的数据结构见表 1，直线与多义线组值组码见表 2。

表 1　直线与多义线数据结构

| 直线 | 多义线 |
|---|---|
| 0 | 0 |
| LINE | LWPOLYLINE |
| 5 | 5 |
| 2B | 2C |
| 8 | 8 |
| 0 | 0 |
| 6 | 100 |
| CENTER | AcDbLine |
| 100 | 70 |
| AcDbLine | 0 |
| …(其他实体组码与组值) | 43 |
| | 0 |
| | …(其他实体组码与组值) |

表 2　直线与多义线组值组码

| 直线 | | 多义线 | |
|---|---|---|---|
| 组码 | 组值 | 组码 | 组值 |
| | | | |
| 0 | 实体类型名 | 0 | 实体类型名 |
| 5 | 实体描述字 | 5 | 实体描述字 |
| 6 | 线姓名 | 8 | 用 ASCII 表示的层名 |
| 8 | 用 ASCII 表示的层名 | 42 | 拱度 |
| 10,20,30 | 直线起点(x,y,z) | 43 | 宽度常数 |
| 11,21,31 | 直线终点(x,y,z) | 10,20 | 顶点坐标 |
| 100 | 子类标志(AcDbLine) | 70 | 多义线标志(位代码) |
| | | 100 | 子类标志(AcDbLine) |

## 2　DXF 数据处理

由上述内容可知，要获取轴类件的三维数据模型，只需通过提取对称的 1/2 二维轮廓曲线并进行绕轴旋转构造整个轴类件曲面即可。通常，轴类件轮廓一般由直线、圆弧等解析形式的二次曲线和有理 B 样条构成。对提取轮廓曲线的问题就转化为如何正确表达和提取上述的这些直线、圆弧以及 B 样条曲线。现对轮廓的数据处理进行如下说明：1>为简化轴类件三维数据模型，现只取 1/2 轴类零件的轮廓曲线和旋转轴；2>需要处理的轴类零件二维轮廓曲线均由直线、圆弧构成；3>采用直线插补的方法对 B 样条曲线进行处理，将其转化为由允许精度下多条直线段共同构成。

另外，数据处理还需考虑如下实际问题。

（1）在实际应用中，图形零件有时会出现轴线不水平的现象。这时，如果直接对零件图形轮廓进行旋转成型，会导致旋转成型的数据处理过程变得十分复杂。为此，就需要使用曲线变换算法对轮廓数据进行预处理，使轴线处于某一特定的位置（本研究取水平 $X$ 轴）。该曲线变换算法的数学计算过程如下：

曲线变换如图 1 所示。已知轴线起、止点和线外任一点坐标分别为（$a$，$b$）、（$c$，$d$）和（$x_0$，$y_0$），对轴线进行平移和回转操作，使轴线变换为起于原点的水平线，求点 $P$ 的坐标。

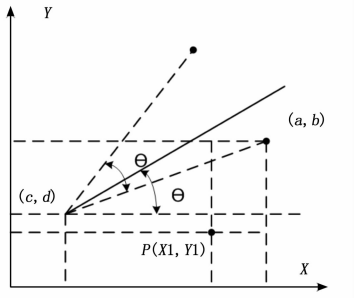


图 1　曲线变换

由平面变换矩阵得：

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_0 - a \\ y_0 - a \end{bmatrix} \tag{1}$$

其中

$$\theta = \arctan\left(\frac{d-b}{c-a}\right) \tag{2}$$

$$\begin{cases} t_1 = x_0 - a \\ t_2 = y_0 - b \end{cases} \tag{3}$$

解得

$$\begin{cases} x_1 = t_1\cos\theta + t_2\sin\theta \\ y_1 = -t_1\sin\theta + t_2\cos\theta \end{cases} \tag{4}$$

（2）圆弧切割问题：

圆弧切割如图 2 所示。在 DXF 文件中，圆弧信息可用起点、终点和对应圆弧拱度定义和描述。在进行后续横向切割时，需要根据已知信息求出对应圆弧的圆心坐标，相关算法介绍如下：

$$\text{令 } l = \frac{\sqrt{(c-a)^2 + (d-b)^2}}{2} \tag{5}$$

由拱度的定义可知 $bow = \dfrac{h}{l}$，

所以

$$h = l * bow \tag{6}$$

由图中三角关系可得：

$$r^2 - l^2 = (r-h)^2 \tag{7}$$

可得圆弧半径 $r = \dfrac{h^2 + l^2}{2 \times h}$ 。

令 $\omega = \arctan\left(\dfrac{d-b}{c-a}\right)$

则弧线圆心坐标为：

$$\begin{cases} x_0 = \dfrac{a+c}{2} - (r-h)x\sin\omega \\ y_0 = \dfrac{b+d}{2} + (r-h)x\cos\omega \end{cases} \tag{8}$$

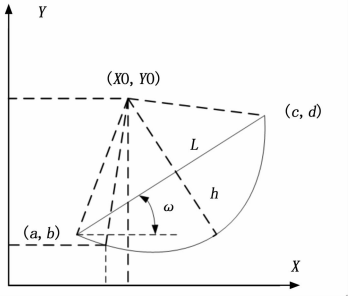


图 2 圆弧切割

(3) 直线插补处理有理 B 样条曲线：

有理 B 样条曲线主要用来描述具有普遍意义的解析曲线，在实际工程中已广泛应用。对于有理 B 样条曲线的数据处理，关键在于提取有效分割点。本应用程序中采用直线插补的方式，对有理 B 样条曲线进行分割点插入，从而可以通过分割点组成的连续直线段高度趋近代替光滑的 B 样条曲线。但在此过程中，关键之处在于需要判断位置误差是否小于允许误差(本应用程序允许的位置误差为 0.1 mm)，否则，则不能进行上述近似替代。

曲线插补计算示意图如图 3 所示，已知 $P1$ ($x_1$, $y_1$, $z_1$)，$P2$ ($x_2$, $y_2$, $z_2$)，$P3$ ($x_3$, $y_3$, $z_3$)，$L$ 为轨迹曲线上点到插补直线段之间的最大距离，即插补误差。当分割点生成后，每两个分割点之间的插补直线均要进行一次误差计算，如果插补误差 $L$ 小于给定的允许误差，则能采用插补直线段来替代轨迹曲线；反之，则必须通过增加分割点的数量来减少误差，直到满足插补误差要求[10]，最后提取满足条件的直线段起始点坐标信息即可。


图 3 插补计算示意图

$L$ 计算公式：

$$L = \frac{|\overrightarrow{P_1P_3} X \overrightarrow{P_1P_2}|}{|\overrightarrow{P_1P_2}|} = \sqrt{\frac{(am-bl)^2 + (bn-cm)^2 + (cn-am)^2}{l^2+m^2+n^2}} \tag{9}$$

其中：

$$\overrightarrow{P_1P_2} = (l,m,n) = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$$

$$\overrightarrow{P_1P_3} = (a,b,c) = (x_3 - x_1, y_3 - y_1, z_3 - z_1)$$

计算 $L$ 的程序语句为：

l=sqrt (pow ( (x1−x2), 2) +pow ( (y1−y2), 2));

(4) 考虑到三维数据模型的曲面光滑度，需要设置圆弧切割密度。这里分为横向切割和纵向切割，其中横向切割是指在已知轮廓线上逐段均匀地分割成若干个点，把所有分割点坐标按次序存入所设置的 3 个二维数组 $x$ [$i$, $j$]，$y$ [$i$, $j$]，$z$ [$i$, $j$] 的过程。程序中横向分割点的个数即为横向切割密度值，值越大，则生成的曲线轮廓就越接近于原图形曲线轮廓，与此同时数据的计算量也会相应增大；纵向切割要对横向切割生成的全部点进行空间三维旋转操作，还需对每点的圆周轨迹进行插值点的坐标计算。插值点数就是密度值大小，即纵向插值点数越多，分割越细，所生成真实感图形横截面越圆滑，但数据的计算量也相应增大。由于算法原理相同，现主要介绍纵向切割，如下所示：

圆弧纵向切割如图 4 所示。由横向切割得到母线轮廓的全部坐标点数组 $x$ [$i$, $j$]，$y$ [$i$, $j$]，$z$ [$i$, $j$]，进行纵向切割计算时，假设绕 $x$ 轴作圆周回转操作，要计算圆周轨迹上的 $n$ 个点的坐标，设旋转步长 $t$ 为 $2\pi/n$，可得旋转变换矩阵 $\boldsymbol{T}$：

$$\boldsymbol{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & -\sin t \\ 0 & \sin t & \cos t \end{bmatrix} \tag{10}$$

设母线上任一点坐标为 ($x$, $y$, $z$)，则旋转后的点坐标为：

$$[x* \quad y* \quad z*] = [x \quad y \quad z] * \boldsymbol{T} \tag{11}$$

即：

$$\begin{cases} x* = x \\ y* = y\cos t + z\sin t \\ z* = -y\sin t + z\cos t \end{cases} \tag{12}$$

其中 $z=0$，相关程序算法如下：

```
for(int j=1;j<m_m;j++)//取多少份;
{ for(inti=0;i<ver_num;i++)//横向总点数;
{ verx. Add(verx. GetAt(i));
  very. Add(very. GetAt(i) * cos(j * 2 * PI/m_m));
  verz. Add(very. GetAt(i) * sin(j * 2 * PI/m_m));}}
```

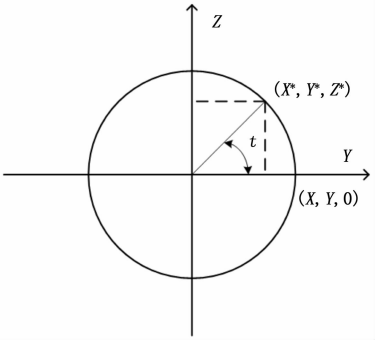


图 4 圆弧纵向切割

(5) 判断渲染方向：

OpenGL 中光照处理涉及到平面法向量。平面法向量决定了光照的正反面，只有获得所期望的法向量才能使光照得到有

效的处理。为此，需要单独设计子程序模块来计算和确定平法向量。

已知，不共线的 3 个点确定一个平面，两两不同的点确定一个矢向量，而不同矢向量的叉积便能够确定平面法向量的大小和方向。平面法向矢量如图 5 所示。

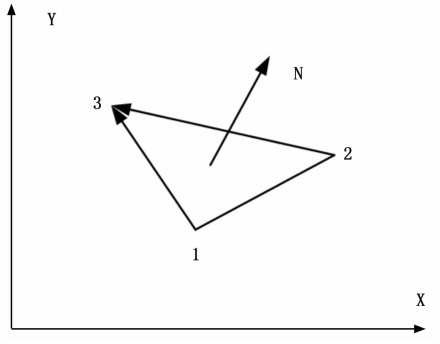


图 5　法向矢量

在空间坐标系中，设 1、2、3 点的坐标值分别为 $(x_1, y_1, z_1)$，$(x_2, y_2, z_2)$，$(x_3, y_3, z_3)$，则所构成的平面法向量的计算过程如下：

采用空间解析几何的方法求得矢量 $\overrightarrow{13}$ 和 $\overrightarrow{23}$，那么平面法向量的空间描述为：

$$\begin{vmatrix} i & j & k \\ x_{13} & y_{13} & z_{13} \\ x_{23} & y_{23} & z_{23} \end{vmatrix} \tag{13}$$

$$\begin{cases} x13 = x1 - x3 \quad , \quad y13 = y1 - y3 \quad , \quad z13 = z1 - z3 \\ x23 = x2 - x3 \quad , \quad y23 = y2 - y3 \quad , \quad z13 = z1 - z3 \end{cases} \tag{14}$$

矩阵式交叉展开，可以得到法向量对应的数学表达式：

$$(y_{13}z_{23} - z_{13}y_{23})i + (z_{13}x_{23} - z_{23}x_{13})j + (x_{13}y_{23} - x_{23}y_{13})k \tag{15}$$

经过上述几种算法的处理，基本上就能实现 DXF 轴类零件图形数据的处理，将处理后的相关数据保存在所设置的动态数组中，以便为后续真实感渲染做好准备。

## 3　DXF 数据读取

要读取 DXF 文件中图形数据，首先打开要读取的 DXF 文件，分别读取对应图元的组码和组值。当读到组值等于“ENTITES”时，表明实体段开始，随后读取的第一个组值即为图元类型。然后判断该图元类型，根据对应的图元类型，提取该图元信息并保存到已经设置的动态数组中。就这样反复读取，直到组值为“ENDSEC”时，表示实体段读取结束，即几何图元的实体信息提取完毕[11]。DXF 文件的数据读取流程如图 6 所示。

## 4　实现真实感

系统程序利用 OpenGL 主要实现以下功能：窗口初始化，图形的绘制，坐标变换，光照和材质设置，雾化和纹理贴图等功能[12]。由于涉及多种 OpenGL 技术，现主要介绍与真实感显示贴切的光照与材质设置内容，并以透视投影变换为例，介绍其变换原理。

### 4.1　光照与材质设置

现实世界中的物体都具有多种多样的色彩和质感。概括来

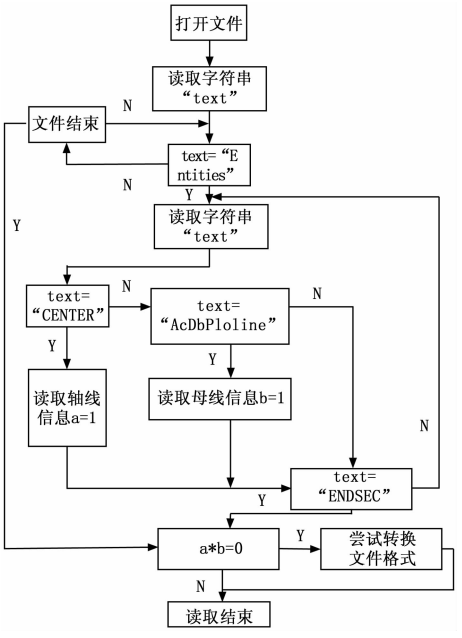


图 6　DXF 文件的数据读取流程

说，物体的颜色是由其本身的光学特性与照射到它身上的外来光共同作用的结果。在 OpenGL 中，可利用光源和物体的材质来模拟自然界的光学效果，其中的光源用来模拟外来光，材质用于描述物体本身的光学特性。

（1）颜色。

彩色计算机屏幕可以显示各种颜色，而实际上它们都是有红、绿、蓝三色光以不同比率组合而成的。OpenGL 提供了 RGBA 与颜色索引表两种描述颜色的方法。为充分利用 OpenGL 强大的渲染功能，本文选用 RGBA 模式，且初始时将零件实体的颜色设置为宝石蓝（如实验结果中的图 9 所示）。

（2）光源。

光源就是可以发光的物体，它同一般的几何体一样受到几何变换矩阵的影响。OpenGL 利用有限多个光源来尽量模拟自然的光效。对于一个光源来说，其属性主要有环境光、泛光、镜面光、光源位置、聚光灯方向、光锥角度及三级衰减因子等，在使用时可以根据需要对各参数进行调整。

本真实感应用中设置光源的代码如下：

```
void CThirdEyeView:: OnSetLight ()
{ CPropertySheet SetLight (" 灯光设置");
  SetLight. m_psh. dwFlags | = PSH_NOAPPLYNOW;
  CDLG_SETLIGHT dlg [8];
  for (int i=0; i<8; i++) //根据不同的灯号取用不同的灯光效果;
   { SetLight. AddPage (&dlg [i]);
   dlg [i]. Light=Light [i];
dlg [i]. m_Num. Format (" 灯光 %d", i+1);}...}
//上述 8 种灯光设置;
glEnable (GL_LIGHTING);
for (int i=0; i<8; i++)
{glLightfv (GL_LIGHT0+i, GL_AMBIENT, Light [i]. ambient);
……;
if (Light [i]. bIfEnable) glEnable (GL_LIGHT0+i);
```

```
else glDisable (GL_LIGHT0+i);}
```

(3) 材质。

在 OpenGL 中，物体的材质是通过对光的红、绿、蓝三色的反射率定义的。通过设置材质属性，我们才能够识别三维实体模型是由什么构成。真实感中，设置物体材质的代码如下：

```
float tt1 [4], tt2 [4], tt3 [4], tt4 [4]; //RGBA;
for (int i=0; i<4; i++)
 { tt1 [i] =Material. ambient [i] /255.;
tt2 [i] =Material. diffuse [i] /255.;
tt3 [i] =Material. specular [i] /255.;
tt4 [i] =Material. emission [i] /255.;}
glMaterialfv (GL_FRONT, GL_AMBIENT, tt1);
glMaterialfv (GL_FRONT, GL_DIFFUSE, tt2);
glMaterialfv (GL_FRONT, GL_SPECULAR, tt3);
glMaterialfv (GL_FRONT, GL_EMISSION, tt4);
glMaterialf (GL_FRONT, GL_SHININESS, Material. shiniess);
```

OpenGL 通过物体的材质和光源设置，便可准确地模拟现实世界的光照效果，从而实现计算机上三维实体模型的真实感效果。

### 4.2 透视投影[13]

坐标 $P$ 点到观察平面上点（$x_p$，$y_p$，$z_{vp}$）的透视投影如图 7 所示。

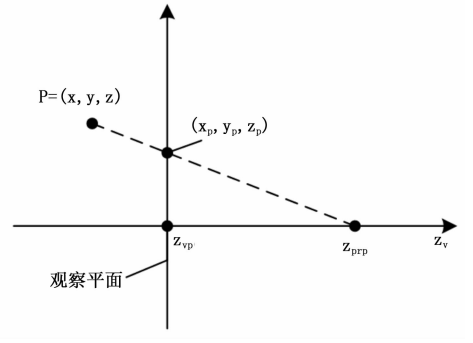


图 7 $P$ 点到观察平面上点（$x_p$，$y_p$，$z_{vp}$）的透视投影

将透视投影按照齐次矩阵的形式记为：

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ h \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{z_{vp}}{d_p} & z_{vp}\left(\frac{z_{prp}}{d_p}\right) \\ 0 & 0 & -\frac{1}{d_p} & \frac{z_{prp}}{d_p} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{16}$$

其中：$d_p = z_{prp} - z_{vp}$ //投影参考点到观察平面的距离；

$h = \frac{z_{prp} - z}{d_p}$ //齐次因子；

可得投影点坐标 $\begin{cases} x_p = \frac{x_h}{h} \\ y_p = \frac{y_h}{h} \end{cases}$ (17)

其中，投影坐标中 $z_p$ 值保持不变。

另外，调用 OpenGL 图形操作函数：glTranslate（平移）、glRotate（旋转）、glScale（缩放）等亦可实现对零件图形的其它操作。

## 5 实验结果

运行程序，首先读入简化后的 1/2DXF 轴类测试件。经过旋转造型后，获得的零件三维立体效果，初始设置为宝石蓝，图 8 为旋转过程中的视图一览。

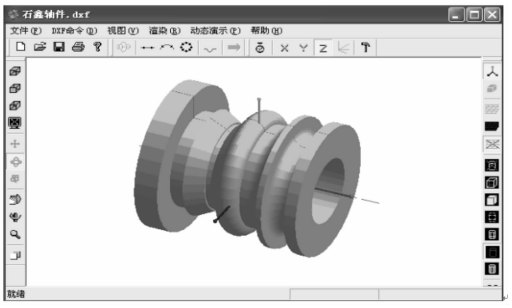


图 8 旋转视图

另外，该真实感应用程序亦可实现反走样、纹理贴图等功能。通过测试，程序运行稳定，DXF 轴类件轮廓数据提取可靠，具备高效的图形真实感显示效果。

## 6 结束语

本研究主要对 DXF 车削轴类零件图形的真实感显示进行研究，通过深入分析 DXF 数据结构，采用曲面构造算法处理与读取 DXF 图形信息，并结合 OpenGL 图形渲染方面的知识对生成的三维模型进行数据处理，完成了对车削轴类零件的真实感显示内容，实现了零件材质、光照、雾化等特殊处理功能。

(1) 解决了计算机辅助车削加工系统中的轴类件轮廓的数据提取问题，编程实现一些数学算法并将这些算法应用于提取和处理轴类件轮廓中，避免了旋转造型生成三维数据时繁琐的数据处理，充分应用了软件处理图形的能力，具备科学计算的特点。

(2) 真实感显示不仅是计算机图形学在计算机领域中的一种实际应用，而且是对文件图形数据挖掘与可视化相结合的一种体现。为更好地应用计算机图形处理技术，这就要求今后在计算机体系结构和计算机算法的本质方面作更深层的研究。

**参考文献：**

[1] JanBender，KennyErleben，Jeff Trinkle. Interactive Simulation of Rigid Body Dynamics in Computer Graphics [J]. Computer Graphics Forum，2014，33 (1)：246－270.

[2] 梁秀霞，韩慧健，张彩明. 基于物理仿真的布料动画研究综述 [J]. 计算机研究与发展，2014，51 (1)：31－40.

[3] 刘庆元，易柳城，刘 莉. 基于 diamond－square 算法的数字地形模型构建与三维可视化研究 [J]. 测绘工程 2014，23 (2)：1－4.

[4] 唐 云，罗俊松. 计算机图形技术在数据计算方面的应用 [J]. 制造业自动化，2010，32 (11)：198－200.

[5] BoulangerK，Pattanaik S N，Bouatouch K，Rendering grass in real time with dynamic lighting. [J]. IEEE Computer Graphics and Applications，2009，29 (1)：32－41.

[6] Li C F，Guo X Y，Lu S L，et al. Real－time simulation of meadow [A]. Proceedings of the 7th International Conference on System Simulation and Scientific Computing [C]. Washington D C：IEEE Computer Society Press，2008：1205－1207.

[7] 王 敏，邵定宏. 图形文件格式兼容性研究与发现 [J]. 计算机

工程与设计，2007，28（15）：3758－3761.
[8] 徐呈艺，刘　英，焦恩璋，等. 基于 DXF 文件工业机器人作业程序的生成［J］. 现代制造工程，2014（8）：36－40.
[9] 李洪声. 数控铣床图形自动编程系统设计中 DXF 文件图元信息读取技术研究［J］. 科学技术与工程，2010，10（33）：8241－8246.
[10] 武传宇，贺磊盈，李秦川，等. 基于 CAD 模型的鞋底喷胶轨迹生成方法［J］. 计算机辅助设计与图形学学报，2008，20（5）：678－682.
[11] 梁海涛，马军林，童创明，等. 基于 DXF 模型的数据读取与分析方法［J］. 空军工程大学学报（自然科学版）2007，8（2）：46－48.
[12] 尹海峰，库祥臣. 基于 Visual C＋＋与 OpenGL 的风电机组仿真系统研究［J］. 组合机床与自动化加工技术，2013（6）：142－144.
[13] Donald Hearn M. Pauline Baker. 计算机图形学（蔡士杰，吴春鳗，孙正兴，等译）［M］. 北京：电子工业出版社，2002.

（上接第 255 页）

心相对于支撑对角线滞后 $s/6$ 时基本能够实现稳定的对角小跑。

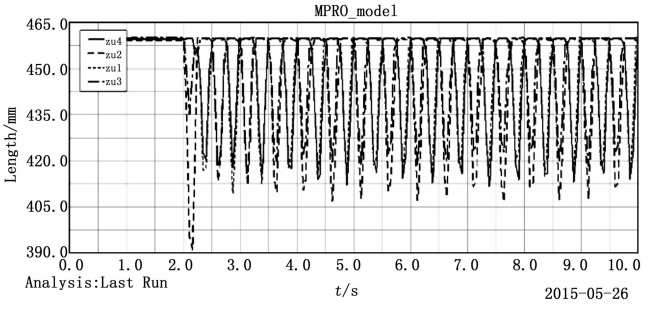


图 7　四足足端竖直方向运动轨迹

从图 8 和图 9 中可以看出随着 2 s 后开始运动，随着速度的不断增加到预定速度，摆动角逐渐趋于稳定，机体的俯仰角在速度稳定后也程周期性波动范围为（－1°，0.5°），基本可以看做机体与地面保持平行。由仿真实验可以看出采用支撑足初始前置 $\frac{s}{2}$，初始重心相对于支撑对角线滞后 $\frac{s}{6}$ 的对角步态时，摆动角能够实现在一定范围内稳定的运动，并可以看出对足在落地时刻的竖直基本重合，表明能够实现稳定对角小跑。

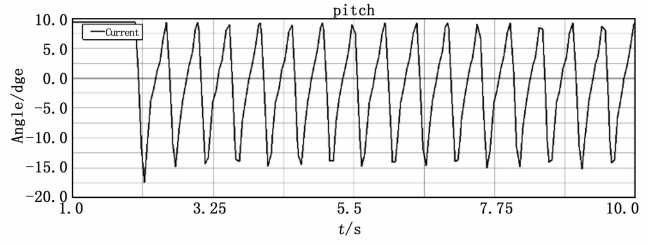


图 8　机器人支撑腿摆动角度变化曲线

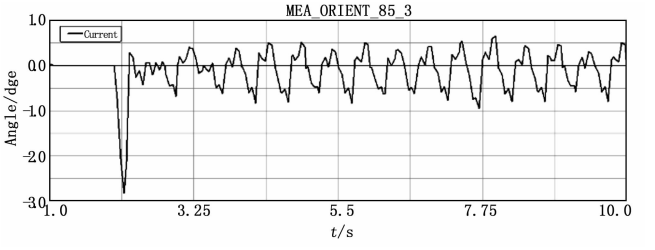


图 9　机器人机体俯仰角变化曲线

## 4　结论

本文通过建立匀速对角小跑对支撑对角线的角动量方程，经过简化得到摆动方程。从仿生学的角度确定了初始支撑足的位置，根据摆动方程以摆动足的同时着地为出发点确定了初始重心位置与步长的关系。在机器人结构尺寸，质量分布等确定的情况下，对对角步态的允许速度范围进行讨论，分析了机构参数对对角小跑速度的影响。通过仿真实验表明在对角小跑过程中，机器人能够实现稳定的对角小跑，摆动角在一定的范围内稳定摆动。

**参考文献：**

[1] Inagaki K，Kobayashi H. Dynamical motion control for quadruped walking with autonomous distributed System［A］. Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems［C］. 1994（2）：1004－1010.
[2] 何冬青，马培荪，曹　曦，等. 四足机器人对角小跑起步姿态对稳定步行的影响［J］. 机器人，2004，26（6）：529－532.
[3] 何冬青，马培荪，曹　曦，等. 四足机器人对角小跑起步姿态对稳定性的影响［J］. 上海交通大学学报，2005，39（6）：880－883.
[4] 刘　蕊，俞志伟，王　鹏，等. 仿生四足机器人对角步态规划及稳定［J］. 科学技术与工程，2013，12（36）：10852－10856.
[5] Schmiedeler J，Siston，R，Waldron，K. The significance of leg mass in modeling quadrupedal running gaits［A］. 14th Symposium on Theory and Practice of Robots and Manipulators［C］. Udine，Italy. 2002（1）：481－488.
[6] 谢惠祥 尚建中 罗自荣 薛勇 四足机器人对角小跑中机体翻转分析与姿态控制［J］. 机器人，2014，36（6）：676－682.
[7] Wensing，Patrick M，Orin，David E. High－Speed Humanoid Running Through Control with a 3D－SLIP Model［A］. IEEE/RSJ International Conference on Intelligent Robots and Systems（IROS）［C］. Tokyo，Japan. 2013（3）：5134－5140.
[8] 胡凌云，孙增圻. 双足机器人步态控制研究方法综述［J］. 计算机研究与发展，2015，16（5）：728－733.
[9] 史耀强. 双足机器人步行仿真与实验研究［D］上海：上海交通大学，2008.
[10] Motoi，Naoki，Suzuki，Tomoyuki，Ohnishi，Kouhei. A Bipedal Locomotion Planning Based on Virtual Linear Inverted Pendulum Mode［J］. IEEE Transactions on Industrial Electronics. Bombay，India：IEEE 2009：54－61.
[11] Schweigart G，Mergner Y. Human stance control beyond steady stare response and inverted pendulum simplification［J］. Experimaental Brain Research，2008，185（4）：635－653.