doi: 10.7690/bgzdh.2016.10.007

针孔点胶的轨迹指令解析

王体泮，张　健
（绵阳市维博电子有限责任公司，四川 绵阳 621000）

摘要：阐述了针孔点胶控制系统中关于点胶轨迹的解析算法，并成功应用于实际产品研发中，从算法层面上解决了点胶过程中拐角控胶问题，并采用多线程的 DLL 方式实现了算法的模块化与封装。

关键词：针孔点胶；拐角控胶；轨迹解析
中图分类号：TP311.5　　**文献标志码**：A

Track Analyzing in Needle Dispense

Wang Tipan, Zhang Jian
(*Mianyang Weibo Electronic Co*., *Ltd*, *Mianyang* 621000, *China*)

Abstract: The article describes the arithmetic of track analyzing in needle dispense control system, and the method is used to development and research of practical production successfully and resolves the problem of corner dispense and uses the multiple thread DLL method to achieve the arithmetic module and package.

Keywords: needle dispense; corner dotting; track analyzing

0　引言

所谓针式点胶指胶水被可控的胶泵从针孔大小的尖端连续不断地挤出，在执行机构的作用下，被挤出的粘性胶水通过针尖沿预定轨迹堆积，满足预期的点胶要求。

针式点胶是多种点胶方式中对轨迹要求最高的一种点胶方式，要求点胶速度快，点胶轨迹精度高，尤其是对轨迹拐角的胶轨迹的厚度、宽度均高于其它点胶场合的要求。

1　点胶轨迹命令

虽然点胶轨迹属于轨迹运动控制范围，但它不同于常规的数控系统，采用 G 代码编程，满足 ISO 标准，而是有自身的一套编程语言。它采用描述性语句来描述，并赋予指定的工艺意义，满足点胶应用需要，具体如表 1 所示。

表 1　点胶轨迹描述及工艺意义

序号	描述符	意　义
1	Dispense Dot	点点胶
2	Line Start	直线开始点
3	Line Passing	直线节点
4	Circle	整圆
5	Arc	圆弧
6	Line End	直线终点
7	End Program	程序或子程序结束
8	CCD Sensor	CCD 校正
9	High Sensor	测高仪测高
10	Dispense On/Off	点胶开/关
11	Goto Address	跳转
12	Step & Repeat *X*	在 *X* 向重复指令段
13	Step & Repeat *Y*	在 *Y* 向重复指令段

续表 1

序号	描述符	意　义
14	Brush Area: Rectangle	矩形刷
15	Brush Area: Circle	圆形刷
16	Brush Area: Rectangle 1	矩形刷 1
17	Brush Area: Rect. Band	矩形带刷
18	Brush Area: Circle. Band	圆形带刷
19	Brush Area: Circle 1	圆弧刷 1
20	Call Subroutine	调子程序
21	Set I/O	设置 I/O 条件
22	Wait Point	等待点
23	Stop Point	停止点
24	Home Point	回零点
25	Loop Address	循环
26	Dummy Point	虚点
27	Initialize	初始经
28	Lable	标签
29	Needle Adjustment	针调整
30	Needle Adjustment Counter	针调整计数器
31	Access PLC_IO	访问 PLCI/O
32	Tip Clean	清胶
33	Line Speed	直线点胶速度
34	Line Dispense Setup	直线点胶设置
35	Point Dispense Setup	点点胶设置
36	Dispense End Setup	点胶结束设置
37	Arc Dispense Time	圆弧点胶时间

2　点胶指令内部表示

```
typedef struct
{
    unsigned short  usLineId;          // 程序段号
    unsigned short  InstructionId;     // 指令编号
    union
    {
    DISPENSEDOTINF          Inf_DispenseDot;
    LINESTARTINF            Inf_LineStart;
    LINEPASSINGINF          Inf_LinePassing;
    CIRCLEGINF              Inf_Circle;
```

收稿日期：2016-06-24；**修回日期**：2016-08-07
作者简介：王体泮(1972—)，男，四川人，大学本科，研高，从事数控技术研究。

```
    ARCINF                  Inf_Arc;
    LINEENDINF              Inf_LineEnd
    ENDPROGRAMINF           Inf_EndProgram;
    CCDSENSORINF            Inf_CCDSensor;
    HIGHSENSORINF           Inf_HighSensor;
    DISPENSEONINF           Inf_DispenseOn;
    DISPENSEOFFINF          Inf_DispenseOff;
    GOTOINF                 Inf_Goto;
    STEPREPEATXINF          Inf_StepReteatX;
    STEPREPEATYINF          Inf_StepReteatY;
    BRUSHAREARECTINF        Inf_BrushArea;
    CALLSUBROUNTINEINF      Inf_CallSubRountine;
    SETIOINF                Inf_SetIO;
    WAITPOINTINF            Inf_WaitPoint;
    STOPPOINTINF            Inf_StopPoint;
    HOMEPOINTINF            Inf_HomePoint;
    LOOPINF                 Inf_Loop;
    DUMMYPOINTINF           Inf_DummyPoint;
    INITIALIZEINF           Inf_Initialize;
    LABLEINF                Inf_Lable;
    DISPLAYCOUNTERINF       Inf_DisplayCounter;
    LOOPCOUNTERINF          Inf_LoopCounter;
    SETGPIOINF              Inf_SetGPIO;
    TIPCLEANINF             Inf_TipClean;
    LINESPEEDINF            Inf_LineSpeed;
    ….
    }EntryInf;      // 不同指令的入口信息的联合体
}PROGNODEINF;// 程序节点信息
```

3　典型轨迹指令语义

3.1　Brush Area: Circle

区域刷—整圆刷。刷一个整圆区域：由 Line start 和 Line End 定义了此圆半径。整圆刷分封闭与非封闭 2 种类型，受控于编程参数，均由指定起点到终点间按线性变圆弧表示，即 $r=k\times\beta$，其中 k 是比例系数，如图 1 所示。

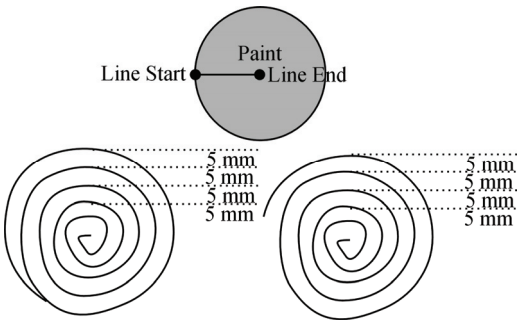

图 1　整圆刷封闭/非封闭

```
typedef struct
{
    double          xs,ys,zs;  // 起点坐标
    double          xe,ye,ze;  // 终点坐标
    unsigned short  usMode;    // 开放或闭合
    double          width;     // 宽度，单位 mm
}BRUSHAREARCIRCLEINF;
```

3.2　Brush Area: Rectangle

区域刷—矩形刷。刷一个矩形区域：由 Line start 和 Line End 定义了此区域范围，如图 2 所示。

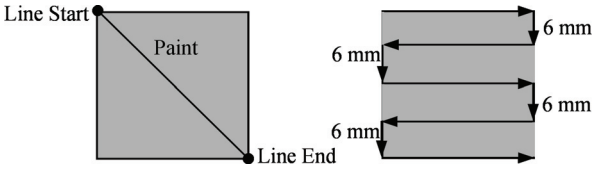

图 2　矩形刷

```
typedef struct
{
    double          xs,ys,zs;  // 起点坐标
    double          xe,ye,ze;  // 终点坐标
    double          width;     // 宽度，单位 mm
}BRUSHAREARRECTINF;
```

3.3　Brush Area: Rect Band

区域刷—矩形带刷。带刷一个矩形区域：由 Line start 和 Line End 定义了此区域范围，如图 3 所示。

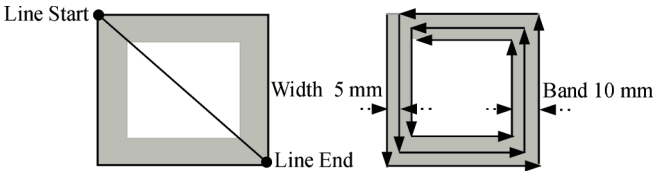

图 3　矩形带刷

```
typedef struct
{
    double          xs,ys,zs;  // 起点坐标
    double          xe,ye,ze;  // 终点坐标
    double          width;     // 宽度，单位 mm
    double          band;      // 带宽，单位 mm
}BRUSHAREARRECTBANDINF;
```

3.4　Step & Repeat *X*

以 *X* 轴为主的程序段循环，有 S 形路径和 N 形路径 2 种循环方式，如图 4 所示。

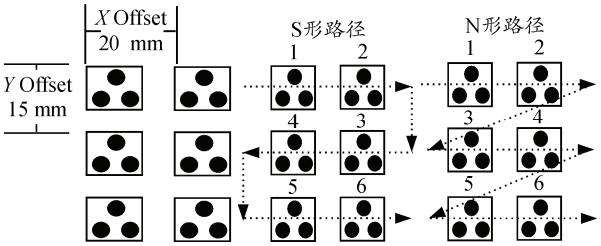

图 4　沿 *X* 向逐段循环

```
typedef struct
{
    unsigned short  usCol;     // 几列
    unsigned short  usRow;     // 几行
    double          Offset_x;  // X 向偏移
    double          Offset_y;  // Y 向偏移
    unsigned short  usAddr;    // 循环地址行号
```

```
    unsigned char   cLable[16];  //循环地址标号
    unsigned short      usPathMode;  //路径模式
}STEPREPEATXINF;
```

类似还有沿 *Y* 向的程序段循环 Step & Repeat *Y*。

4　轨迹解析模块设计

轨迹解析是点胶指令解析中的占据重要地位的分支，涉及到轨迹分类、相邻段过渡转接、轨迹软件粗插补等多个步骤[1]。所有这些复杂、细致的处理过程均被封装在一个叫 DotAnalyse_Dll.Dll 的库中，并将多线程[2]技术应用到该动态库中。

4.1　开发环境

笔者采用 Visual Studio 2008 为开发环境，最后向调用者提供在 WinXP 环境下适用的 DLL，并采用 MFC 进行编程[3]。

4.2　设计思路

基于软件工程的设计思想，笔者采用模块化的设计思想，指导整个软件构架的搭建。

模块间的信息交换主要采用指针参数、少量的全局变量的方式，并采用临界区保护的方式实现各模块间的数据同步[4-6]。设计思路如图 5 所示。

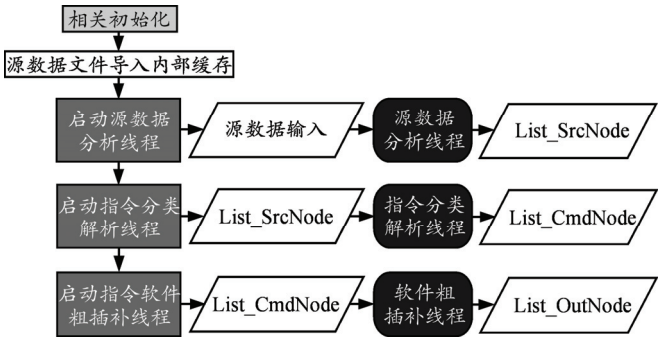

图 5　算法模块原理流程

4.3　算法模块接口

1) bool Module_Start(INF_ENTRY *pInf_Entry);

2) void GetVersion(VERSIONINF *pVersionInf);

算法模块只向调用者提供 2 个接口 API，模块的启动和获取算法模块的版本信息。

在接口函数 Module_Start 的入口信息中，包含着如下复杂的数据结构信息：

```
typedef struct
{
 CString   FileName;// 点胶程序文件名
 HWND   hWnd;    // 调用者窗口句柄
 unsigned short  usWM_MsgId;    // 消息 Id
 unsigned short  usQueueSize_Out; //循环队列容量
.......
 bool   bStopProcess; //强行中止数据处理过程
 CCriticalSection          *pCriticalSection;
                           // 访问的互斥变量指针
 unsigned short            *pusCycQueue_Front_Out;
 unsigned short            *pusCycQueue_Rear_Out;
 ..........
ENTRY_EXITNODE    *pCycQueueBaseAddr_Out;
 double                    *pdArcChordErr;
 SMOOTHINF                 *pSmoothInf; // 平滑信息
}INF_ENTRY; // 数据处理模块的入口信息
//
```

4.4　拐角控胶设计

由于针孔出胶是连续、均匀的，为了保证点胶轨迹的光滑、一致、均匀，则必须执行机构运动匀速，否则就会出现胶液堆积，尤其是在拐角处。如果运动过程减速，出胶的过程则只有关闭或打开 2 种状态，而不会与运动同步减速，所以此时运动的减速会导致胶的堆积。如何保证运动速度在拐角处仍保持合成速度的恒定不变，成为拐角控胶的关键。采用圆弧过渡方式实现相邻轨迹段间的光滑过渡，可以很好地保证拐角处合成速度的恒定，从而从理论上保证拐角控胶的实现。各种转接关系如图 6 所示。

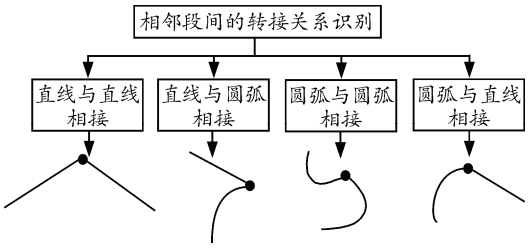

图 6　各种转接关系示意

5　结束语

对于点胶指令的轨迹解析算法已经被成功应用到实际项目中，取得了很好的轨迹控制效果。

参考文献：

[1] 侯俊杰. 深入浅出MFC[M]. 2版. 武汉: 华中科技大学出版社, 2002: 5-10.

[2] 计算机网络与 Internet 教程[M]. 北京: 清华大学出版社, 1999: 11-29.

[3] Microsoft Visual C++ 6.0 类库参考手册(下)[S]. 北京: 北京希望电脑公司, 1999.

[4] 袁震. 基于消息中间件的数据同步更新方法[J]. 兵工自动化, 2013, 32(7): 93-96.

[4] 李璀, 杨贻嘉, 荣嵘, 等. 定时同步算法[J]. 兵工自动化, 2013, 32(12): 23-27.

[5] 陈健, 李晓明, 苏必达, 等. 基于 SFM 的移动机器人运动轨迹的计算方法研究[J]. 机电工程, 2015, 32(11): 1524-1528.