

关于实现Delta结构中等精度低成本3D打印机的项目简介

概述

本项目目的在于构建一台有高度可复制性可定制性的3D打印机，3D打印机结构为Delta结构，在小部件打印上拥有优于同成本笛卡尔结构、SCARA结构和Core/H-Bot结构的打印精度，同时在大型部件打印上有着较快的打印速度。

设计目标

- 打印技术：堆积熔融 Fused Deposition Modeling (FDM)
- 物理结构：Delta
- 控制器：应用于RAMPS 1.4平台的Arduino Mega2560
- 控制固件：Marlin固件
- 运动方式：打印件固定， $\alpha/\beta/\gamma$ 三轴运动
- 速度： $\alpha/\beta/\gamma$ 单轴320mm/s
- 最低分辨率： $\alpha/\beta/\gamma$ 单轴50步/mm
- 最大分辨率： $\alpha/\beta/\gamma$ 单轴300步/mm
- 精度：优于0.3mm (300 μ m)
- 挤出头孔径：0.4mm (可替换)
- 打印层厚：0.05mm-4mm
- 可打印体积：圆柱体，直径170mm，高度240mm
- 框架体积：三棱柱，边长300mm，高度600mm
- 打印表面：可加热直径170mm圆形玻璃，不可移动
- 挤出机结构：远端挤出
- 末端执行器质量：小于50克
- 硬件成本：低于1000 ¥ [RMB]
- 自动水平校准（自动调平）

项目分析

算法

本项目所使用的Delta结构是一种rostock结构变体，这是一种通过3个空间平行并联臂进行运动的结构。

通过将笛卡尔(正交)坐标系运动参数转换为并联臂高度参数，对末端执行器进行移动，达到打印3D物品的目的。

项目难点在于将工业上常用的 GCode 的笛卡尔坐标运动参数值转换为Delta运动参数。

在这里，我尝试使用了以下算法对坐标进行转换：

直接使用直线插值来减少运算量 😊

```
delta[X_AXIS] = sqrt(delta_diagonal_rod_2
    - sq(delta_tower1_x-cartesian[X_AXIS]))
    - sq(delta_tower1_y-cartesian[Y_AXIS]))
    + cartesian[Z_AXIS];
```

框架

本项目使用欧标铝型材作为整体框架的构建材料，对材料方面较为宽容，且整体结构强度高，拆卸简易快捷，可便携化使用。

同时，顶角连接件采用3D打印件，大大增加了磨损部件的可更换性。

CAD设计及模型切片

本项目采用 *OpenSCAD* 作为核心计算机辅助设计工具，*OpenSCAD* 是一个“非主流”的CAD软件。

通过代码生成模型的形式，大大提高了程序的操作性，不必费心费力去学习各种云里雾里的CAD软件。

同时，*OpenSCAD* 是一个开源软件，可以根据自身需要修改编译，基于GPL v3许可证。

通过简单高效的 *OpenSCAD* 脚本去绘制3D模型：

<p>句法</p> <pre>var = value; module name(...) { ... } name(); function name(...) = ... name(); include <...scad> use <...scad></pre>	<p>转换</p> <pre>translate([x,y,z]) rotate([x,y,z]) scale([x,y,z]) resize([x,y,z],auto) mirror([x,y,z]) multmatrix(m) color("colorname",alpha) color([r,g,b,a]) offset(r delta,chamfer) hull() minkowski()</pre>	<p>数学的</p> <pre>abs sign sin cos tan acos asin atan atan2 floor round ceil ln len let log pow sqrt exp rands min max</pre>	<p>功能</p> <pre>concat lookup str chr search version version_num norm cross parent_module(idx)</pre>	<p>其他</p> <pre>echo(...) for (i = [start:end]) { ... } for (i = [start:step:end]) { ... } for (i = [...],...) { ... } intersection for(i = [start:end]) { ... } intersection for(i = [start:step:end]) { ... } intersection for(i = [...],...) { ... } if (...) { ... } assign(...) { ... } import("...stl") linear_extrude(height,center,convexity,twist,slices,scale) rotate_extrude(angle,convexity) surface(file = "...dat",center,convexity) projection(cut) render(convexity) children([idx])</pre>
<p>2D</p> <pre>circle(radius d=diameter) square(size,center) square([width,height],center) polygon([points]) polygon([points],[paths]) text(text, size, font, halign, valign, spacing, direction, language, script)</pre>	<p>布尔运算</p> <pre>union() difference() intersection()</pre>	<p>列表理解</p> <pre>Generate [for (i = range(list) i] Conditions [for (i = ...) if (condition(i)) i] Assignments [for (i = ...) let (assignments) a]</pre>	<p>特殊变量</p> <pre>\$fa 最小角度 \$fs 最小尺寸 \$fn 片段数量 \$st 动画步骤 \$Vpr 视角旋转角度（度） \$Vpt 视口翻译 \$Vpd 视口相机距离 \$children 模块孩子的数量</pre>	<p>修饰符号</p> <pre>* 禁用 _ 只显示 # 突出显示/调试 % 透明背景</pre>
<p>3D</p> <pre>isphere(radius d=diameter) cube(size, center) cube([width,depth,height], center) cylinder(h,r d,center) cylinder(h,r1 d1,r2 d2,center) polyhedron(points, triangles, convexity)</pre>				

本项目使用 *Slic3r* 进行模型切片，即将3D模型文件 .stl 转换为3D打印机固件可以识别的 .gcode 文件。

Slic3r 的软件可定制性非常高，可根据需要及算法需求生成必要的支撑结构，打印件打印完后通过后期减材加工的方式去除。虽然每个版本都有一定的Bug，但不影响使用。

同时，*Slic3r* 是reprap项目的开源软件，根据AGPLv3许可证授权。

末端执行器

整个末端执行器采用散热性能更高的 *H* 散热鳍片结构，更好的降低打印时拉丝的可能性。

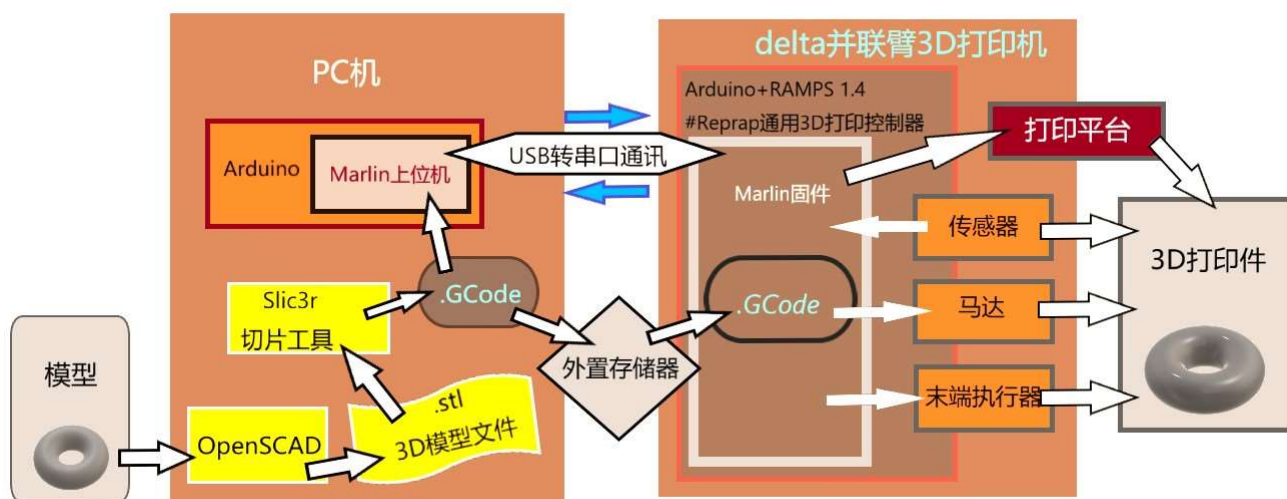
末端执行器整体结构使用改进型的 E3D V6 挤出头结构，在原有E3D v6挤出头的基础上增加了一组光电传感器，在断丝时提供自动暂停打印的功能。

同时，挤出机将使用远端挤出的方式，大大降低了末端执行器的重量，使得整体打印精度提升。

挤出机

本项目使用创新改进型Titan挤出机结构，在原本Titan挤出机结构的基础上通过增加一组从动轮以提高打印机送丝稳定型，降低打印机断丝的几率。

工作流程图



应用方向

FDM工艺下，个人组装设计的3D打印机在实际应用中可以作为中低精度需求模型及快速工件打印，特别是Delta结构下，较小的打印件会有较高的打印精度，同时得益于远端挤出机，3D打印机在隔空层打印时拉丝现象会有较大改善。

总结

目前状况下，3D打印技术更倾向于金属烧结成型、光致成型树脂、生物材料等方向，但是，作为最基础的堆积熔融3D打印技术在技术探索及学习实践方面有着其他技术无可比拟的优势，在不同机械构型及打印技术方面也有一定的前瞻性。3D打印机在嵌入式技术中有着重要的意义，可以为学生提供一个综合性、有拓展空间的平台。

这个项目的实现进展及技术工艺内容，
我将会以 GPL3.0 的协议开源在 Github 上。

Edit by Visual Studio Code
Syntax with Markdown