

Arduino+3D 打印

创新电子制作

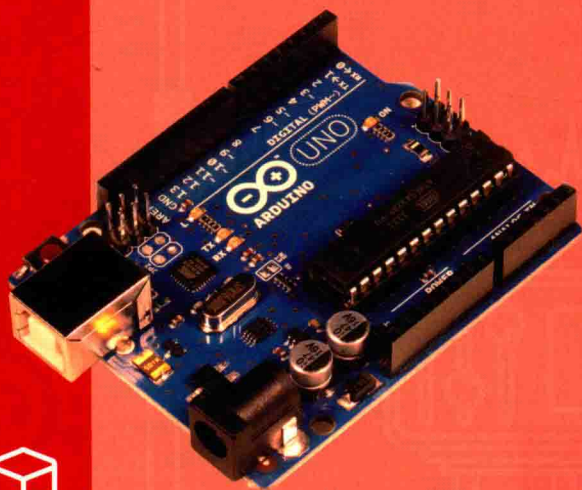


16 个智能硬件搭建实例

陈杰 李岩 著

开源硬件平台 Arduino 与快速成型技术 3D 打印相结合，作品效果大提升！

16 个智能硬件制作项目引领你走上创客之路



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS



DFROBOT
DRIVE THE FUTURE

Arduino+3D 打印 创新电子制作

16 个智能硬件搭建实例

陈杰 李岩 著

人民邮电出版社
北京

I'M MAKER Overlook Pro

图书在版编目(CIP)数据

Arduino+3D打印创新电子制作: 16个智能硬件搭建实例 / 陈杰, 李岩著. — 北京: 人民邮电出版社, 2016.9

(i创客)

ISBN 978-7-115-43213-1

I. ①A… II. ①陈… ②李… III. ①单片微型计算机—应用—电子器件—制作②立体印刷—印刷术—应用—电子器件—制作 IV. ①TP368.1②TS853③TN103

中国版本图书馆CIP数据核字(2016)第192842号

内 容 提 要

“i创客”谐音为“爱创客”，也可以解读为“我是创客”。创客的奇思妙想和丰富成果，充分展示了大众创业、万众创新的活力。这种活力和创造，将会成为中国经济未来增长的不熄引擎。本系列图书将为读者介绍创意作品、弘扬创客文化，帮助读者把心中的各种创意转变为现实。

Arduino是国际流行的开源硬件开发平台，能简单、快速地搭建出智能控制电子制作项目；3D打印是新兴的快速成型技术，能将3D建模软件建立的或通过3D扫描技术获取到的数字化模型制作成实物，非常适合进行可行性试验和小批量制作。将二者结合在一起，我们能制作出拥有像样外形、更像一件工业产品而不是手工DIY作品的电子制作。别再担心作品没有外壳、连线凌乱、需要定制部件了，让3D打印赋予它们精美外观吧。

本书收录了16个Arduino+3D打印制作实例，操作步骤清晰、图片简明、可操作性强，读者既可以直接仿照制作，也可以借此开阔思路，创造出更多、更有创新价值的作品。

◆ 著 陈 杰 李 岩

责任编辑 周 明

责任印制 周昇亮

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号

邮编 100164 电子邮件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

北京画中画印刷有限公司印刷

◆ 开本: 690×970 1/16

印张: 6.5

2016年9月第1版

字数: 141千字

2016年9月北京第1次印刷

定价: 39.00元

读者服务热线: (010)81055339 印装质量热线: (010)81055316

反盗版热线: (010)81055315

广告经营许可证: 京东工商广字第8052号

前言

创客一词源自英文单词“Maker”，是指出于兴趣与爱好，努力把各种创意转变为现实的人。而“创客”一词在刚刚过去的 2015 年在中国大地上着实火了，政府工作报告里首次提到了“创客”一词，指出要让“众多创客脱颖而出”。尽管这只是简短的一句话，但在创客群体中引发了强烈关注。伴随着互联网的迅猛发展，中国的创客团体如雨后春笋般崛起。

而作为创客，必然少不了要应用创客工具，本书以 3D 打印机作为数控加工设备，以 Arduino 控制板作为创客实现创意的工具，通过 3D 打印机打印出机械结构件，再结合 Arduino 进行电子控制，实现具有一定功能的电子互动作品。

什么是 3D 打印？简单点来说，就是快速成型技术的一种，以数字模型文件为基础，运用粉末状金属或塑料等可粘合材料，通过逐层打印的方式来构造物体。

3D 打印的过程通常包括 3D 建模、切片处理、完成打印 3 个步骤，先通过计算机 3D 建模软件建立 3D 模型，再将建成的 3D 模型切分成逐层的截面，即“切片”，从而指导打印机逐层完成打印。

什么是 Arduino？Arduino 是一个开源的、拥有简单输入 / 输出（I/O）功能的电路板，它使用开源 IDE（集成开发环境）。Arduino 可以用来开发独立运作的互动装置。你可在 Arduino 控制板上进行简单的物理电路的搭建，再通过 Arduino IDE 编程来完成一个具有





“生命力”的智能作品。

通过 3D 打印机制作作品机械结构，以 Arduino 来实现作品的功能，你可以构建出个性化的互动作品。

陈杰

2016.5.26

CONTENTS

目录

第1章 3D打印+Arduino

制作初接触 1

01 3D建模+打印个性化钥匙扣..... 2

- 1.1 3D建模..... 2
- 1.2 切片处理..... 5
- 1.3 开始打印..... 8
- 1.4 思路拓展..... 11

02 魔戒与台灯..... 12

- 2.1 作品器材..... 12
- 2.2 制作步骤..... 13
- 2.3 作品测试..... 14

03 3D打印智能自浇花系统..... 16

- 3.1 电容式土壤湿度传感器校准..... 17
- 3.2 电路连接..... 18
- 3.3 程序代码..... 20

04 语音控制3D打印恐龙头骨模型... 22

- 4.1 恐龙头骨机械结构制作..... 22
- 4.2 云台部分组装..... 23
- 4.3 电路连线..... 24

- 4.4 代码编写..... 25

- 4.5 作品测试..... 27

05 3D打印膝关节辅助装置..... 28

- 5.1 工作原理..... 29
- 5.2 制作机械部分..... 29
- 5.3 电子部分..... 31

第2章 制作个性化机器人... 35

06 借助3D打印技术制作4足机器人..... 36

- 6.1 机械结构部分制作步骤..... 36
- 6.2 电子部分制作步骤..... 37
- 6.3 作品总结..... 39

07 3D打印的4足机器人脱机版..... 40

- 7.1 用3D打印机打印出机器人的机械结构..... 40
- 7.2 舵机测试..... 41
- 7.3 零件组装..... 42
- 7.4 安装电池..... 43
- 7.5 初始化舵机位置..... 44

08 用 3D 打印部件制作跳舞**机器人 BOB.....46**

- 8.1 3D 打印模型.....46
- 8.2 机械结构组装.....47
- 8.3 BOB 的电子部分.....48
- 8.4 测试运行.....51
- 8.5 总结拓展.....51

09 让智能手机变身为会走路的**机器人.....53**

- 9.1 部件组装.....54
- 9.2 电路连接.....56
- 9.3 程序代码和手机 App.....56

第 3 章 制作个性化装备..... 57**10 智慧之光——3D 打印钥匙置****物架.....58**

- 10.1 外形设计及 3D 打印.....58
- 10.2 电子部分制作及安装.....60

11 牛角灯帽.....63

- 11.1 3D 打印部分的制作.....64
- 11.2 电路部分安装测试.....64
- 11.3 总装.....67

12 基于蓝牙通信的自行车“驴友”**装备.....69**

- 12.1 机械结构打印.....70
- 12.2 电路结构安装.....70

13 点滴报警系统.....75

- 13.1 振动器.....76

13.2 非接触式液位传感器.....76

13.3 硬件连接.....76

13.4 蓝牙配对.....77

13.5 代码编写.....78

13.6 模型设计及打印.....78

13.7 作品安装.....79

13.8 作品测试.....80

第 4 章 制作物联网装置..... 81**14 通过网络控制的灯.....82**

- 14.1 用户注册.....82
- 14.2 添加设备和传感器.....82
- 14.3 电路连接.....83
- 14.4 加载库文件.....83
- 14.5 烧录程序.....84
- 14.6 制作外壳.....86
- 14.7 测试运行.....87
- 14.8 任务拓展.....88

15 烟雾宝盒——基于 Yeelink 平台的二手烟雾监测报警装置.....89

- 15.1 器材安装.....90
- 15.2 系统设置.....90
- 15.3 代码编写.....92
- 15.4 作品测试.....94

16 DIY 空气净化器.....95

第 1 章

3D 打印 + Arduino 制作初接触



- 01 3D 建模 + 打印个性化钥匙扣
- 02 魔戒与台灯
- 03 3D 打印智能自浇花系统
- 04 语音控制 3D 打印恐龙头骨模型
- 05 3D 打印膝关节辅助装置



3D 建模 + 打印个性化钥匙扣

作为一名 3D 打印爱好者，时常会被朋友要求打印点小玩意，我也很乐意满足朋友的这些小要求。不过最近朋友们不再是简单地下载个模型来打印，而是要——个人定制。这不，有同事买了广本锋范汽车，想让我给她设计一个带有广本标志和她名字缩写的钥匙扣。为了满足同事的要求，我制作了题图所示的个性化钥匙扣并写了这篇教程。



序号	硬件部分名称	数量
1	3D 打印机（笔者用的是 DFRobot Overlord Pro）	1
2	PLA 粉色（直径 1.75mm）打印耗材	1
3	PLA 白色（直径 1.75mm）打印耗材	1
	软件部分名称	
1	Google SketchUp	

1.1 3D 建模

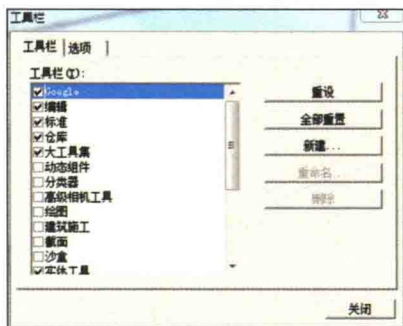
为了降低 3D 建模的门槛，我使用 Google 推出的 SketchUp 进行建模。通过简单的推拉、旋转等操作，即可快速完成建模。

- 1 启动 Google SketchUp，将模版选为“3D 打印 - 毫米”。

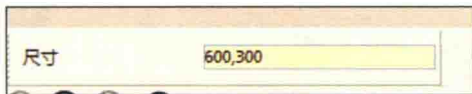


- 2 打开软件，单击“相机”→“标准视图”→“顶视图”，将视图方式调整为俯视图方式。

- 3 单击“视图”→“工具栏”，弹出如下对话框，勾选“大工具集”。



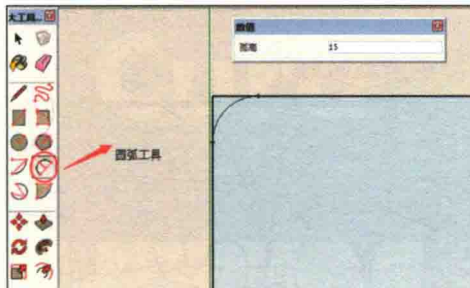
- 4 利用工具栏中的矩形工具绘制一个 $600\text{mm} \times 300\text{mm}$ 的长方体，先单击鼠标确定顶点，然后开始拖动绘制出矩形，此时一定不要进行任何操作，直接在键盘上输入“600,300”，确定后可以定制矩形尺寸。



- 5 使用卷尺工具，在矩形一角距离端点 50mm 处绘制 A、B 两个辅助点。

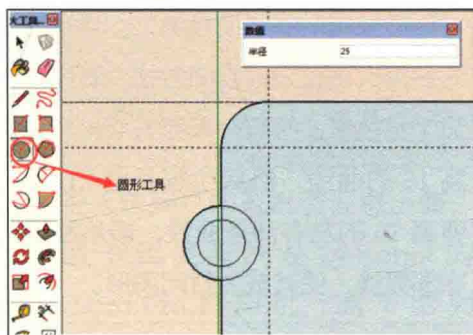


- 6 使用圆弧工具连接两个辅助点，在矩形内生成一个圆角，弧高设置为 15 。

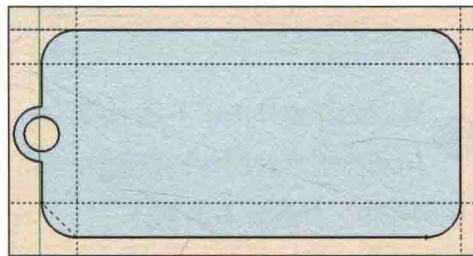


- 7 用选择工具选中圆角外侧的多余部分，按 Del 键删除多余部分。矩形其他三角的操作步骤与之类似，最后要把矩形原本的直角都改为圆角。

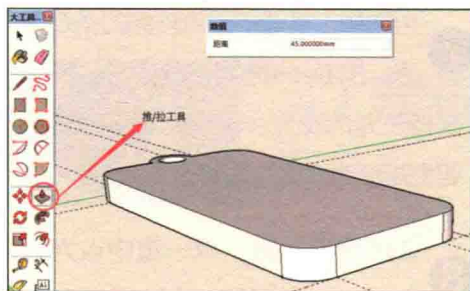
- 8 使用圆形工具，以一边中点为圆心（鼠标指针移动到中点位置时，会有相应提示），分别绘制半径为 40 、 25 的圆形。



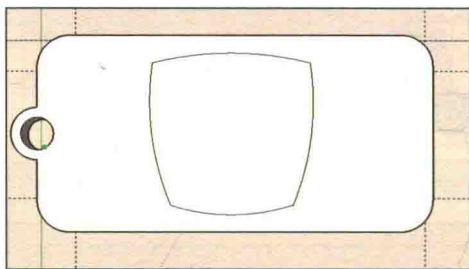
- 9 使用选择工具选中多余部分面积和边线进行删除，如下图所示，完成钥匙扣底板平面的设计。



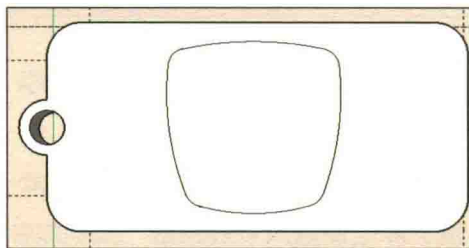
- 10 单击“相机”→“标准视图”→“等轴视图”，切换到等轴视图，使用推拉工具，让钥匙扣底板具有厚度，厚度设置为 45mm 。



- 11** 下面就要在钥匙扣的底板上绘制出车标和姓名了。单击“相机”→“标准视图”→“顶视图”，将视图方式再次调整为俯视图方式。利用圆弧工具在表面绘制一条弧长 240、弧高 15 的曲线，然后绘制长度 210、弧高 15 的左右两条曲线，最后绘制底部弧线，使其成为封闭图形。



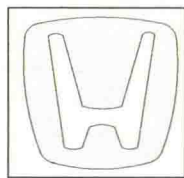
- 12** 这时的标志外形还不是很像，使用圆弧工具对四角进行美化，绘制圆弧角，并删除多余部分。



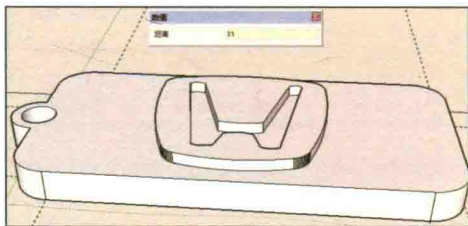
- 13** 使用直线工具、圆弧工具绘制广本车标 H，方法与上面类似，都是一步步修正，效果如下。



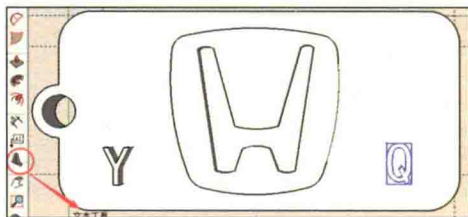
- 14** 选中下图所示的区域，使用推拉工具，进行推拉操作。



- 15** 完成后效果如下。



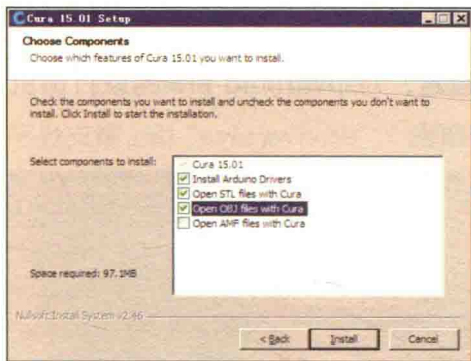
- 16** 使用文本工具输入姓名缩写，这里是 Y 和 Q。分别选中这两个字母，单击鼠标右键，选中分解，并对其进行推拉操作，距离为 15。这样一个简单的个性化钥匙扣的数字模型就做好了。



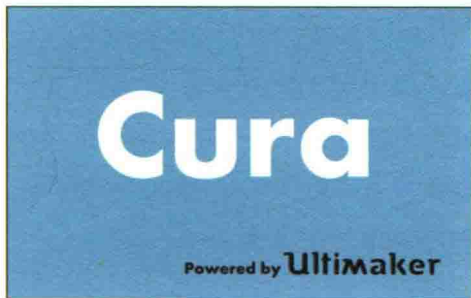
1.2 切片处理

切片处理就是将刚才建立好的3D模型，通过切片处理软件转换成打印机可读取的信息（Gcode）。这里向大家介绍的切片软件是Cura，这是一款用法简单、通用性高的软件。

1 获取安装文件后，双击Cura.exe，进入安装界面，当进入到下图所示界面时，选择自己需要的3D模型文件类型后，单击“Install”按钮进行安装。

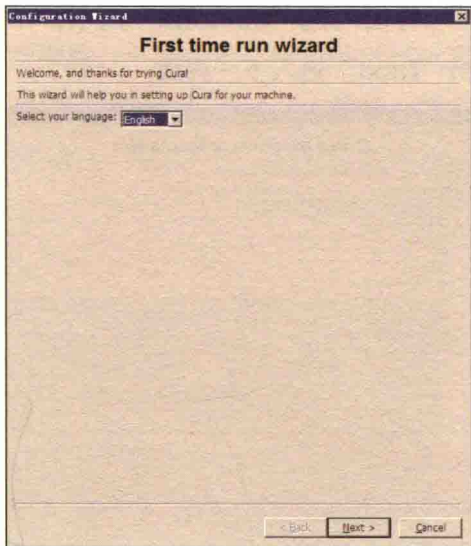


2 此后的安装过程相对简单，直接“Next”一路向下，完成后启动Cura。

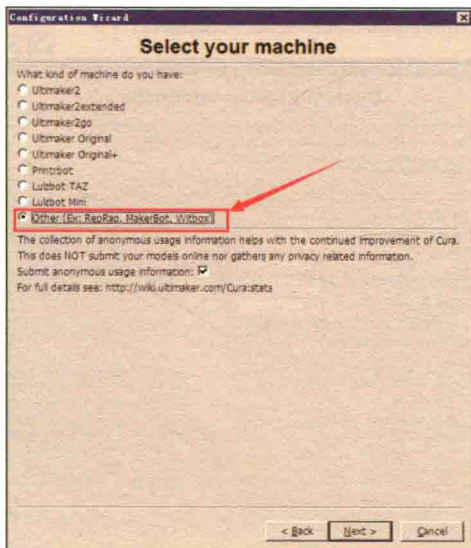


3 第一次运行Cura时会启动向导，选择对应的语言后，单击“Next”

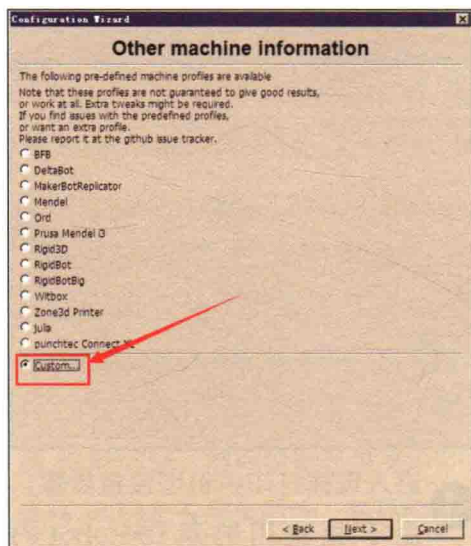
进入下一界面。



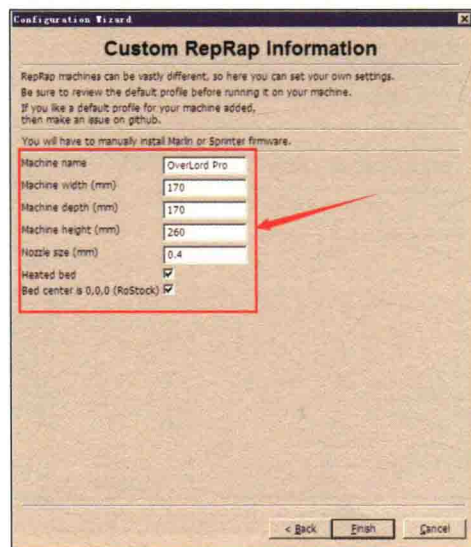
4 进入选择打印机机型设置步骤，由于我使用的是DFRobot的OverLord Pro打印机，而下图列表中并没有提供，所以在机型中选择“other”，单击“Next”进入下一界面。



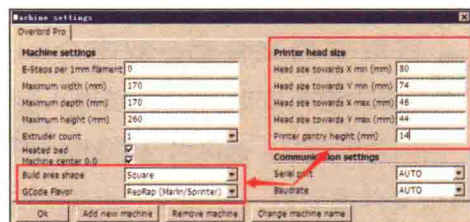
- 5 打开“other machine information”界面勾选“Custom”选项，单击“Next”进入下一界面。



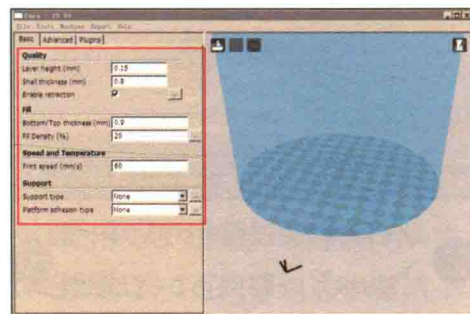
- 6 打开“Custom RepRap information”界面，对机器名称、三轴尺寸、打印喷头尺寸进行设置，根据实际情况勾选加热床和原点位置。



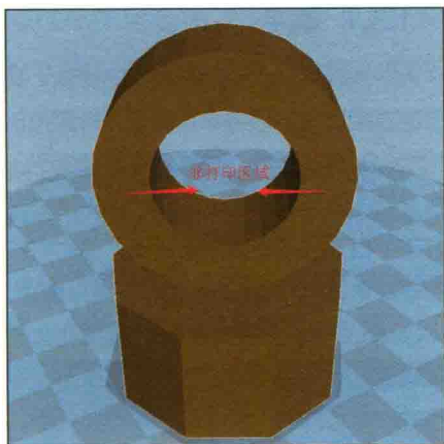
- 7 完成上图初始化配置后，从菜单中选择“Machine”→“Machine settings”，并参照下图键入参数，最后单击“OK”确认。这样就完成了OverLord Pro的参数设置。



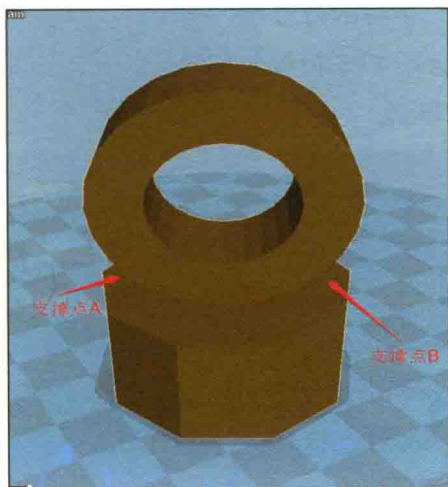
- 8 我们将在以下的步骤中配置打印工作参数。下图为推荐的参数设置，对OverLord Pro这款打印机适用。



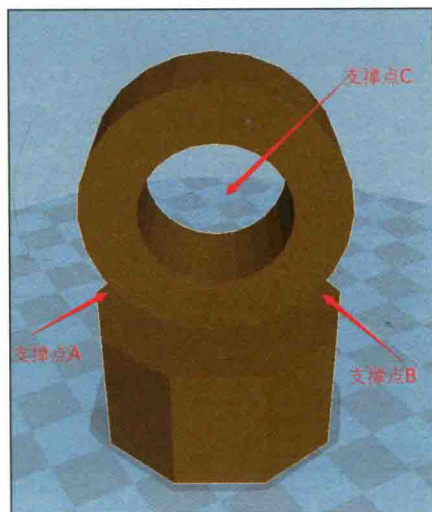
- 9 回缩是指打印过程中，当喷头跨越非打印区域时不吐丝且往回抽丝，以消除“打印区域 A—非打印区域—打印区域 B”打印过程中的拉丝现象。有非打印区域的模型都要勾选启用回抽。



- 10 “Support type”（支撑类型）是指打印有悬空部分的模型时可选的支撑方式，默认为无，另外还包含“touching bulidplate”（接触平台支撑）和“everywhere”（全部）两种类型。选择“接触平台支撑”为部分支撑，系统会为模型与平台之间的悬空部分创建支撑柱。下图开启接触平台支撑后，图中箭头所指的区域就会在打印过程中自动生成支撑柱。



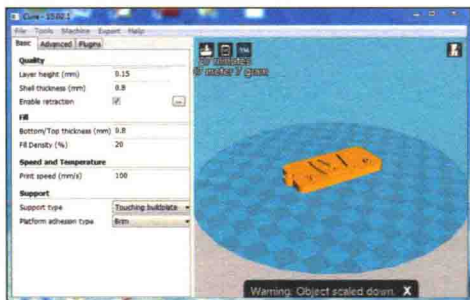
- 11 如果选择“everywhere”支撑类型，模型所有悬空部分都会创建支撑柱。下图中箭头所指的区域就会在打印过程中自动生成支撑柱。通常情况下，考虑到模型后期处理支撑方便，只要对接触的部分添加支撑就可以了。



- 12 “Platform adhesion type”（工作台黏附类型）是指用哪种方式将模型固定在工作台上，默认为“无”。“Brim”是指在模型底层边缘处由内向外创建一个单层的宽边界。“Raft”是指在模型底部和工作台之间建立一个网格形状的底盘，网格厚度可调。“Brim”附着方式较“Raft”易于清除，打印一般选择“Brim”附着。

13 在“Advanced settings”（高级设置）中，用户有更大的自由度来配置打印参数。高级设置中涉及的参数适合高级用户进行调整，如非必要，请不要轻易改变这些参数。

14 用 Cura 导入由 3D 建模软件生成的 STL、OBJ 等格式的 3D 模型文件，格式转换和切片处理会在文件导入 Cura（选择“File”→“Load model file”）后自动完成。在配置参数时，配置所对应的打印时长以及耗材费材料会实时显示在预览窗口的左上角。完成配置后，选择“File”→“Save Gcode”，将 Gcode 格式模型文件保存在 SD 卡中。OverLord 3D 打印机可独立于计算机工作。在将模型保存至 SD 卡中并插入打印机 SD 卡插槽后，即可通过打印机的操作菜单设置并启动打印作业。



1.3 开始打印

对一台机器而言，在通常情况下，3D 打印的过程包括调平、上料、打印。

1.3.1 调平

调平是调整打印初始基准面的步骤，其精准度直接影响打印结果。对于正常用户，在初始化时调平过后，并不需要频繁调平。若在之后的使用过程中出现打印件无法附着至打印平板，或是喷嘴频繁刮擦、撞击打印平板等现象，可以尝试通过重新校准来解决。

OverLord Pro 提供的一键智能化自动调平方案极大降低了调平操作难度，大大提高了打印成功率。注意：在调平之前，请确保打印平板上没有杂物。调平时，请勿将身体伸入机器，以免受到伤害。

1 在主菜单中选择“Calibrate”开始调平，可以选择“AUTO”或“MANUAL”来开始自动或手动调平。



2 选择自动调平时，喷嘴将下降至平台附近进行自动校准，校准完成后，打印机操作菜单会返回至主菜单界面。



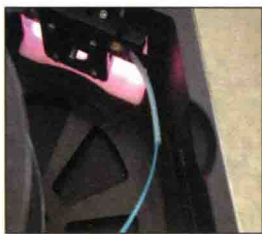
3 选择手动调平时，请确保打印机料管内已加载材料。打印机会在打印平台上打印一个完整的圆圈。打

印过程中可选择“↑”或“↓”来调整喷嘴到平台的高度，直到此圆完整地附着在打印平台上。用户可通过观察喷嘴所画圆圈是否完整附着在打印平台上，来判断平台是否已被调平。当打印机能够正常打印出一个圆圈时，选择“OK”结束调平步骤。



1.3.2 上料

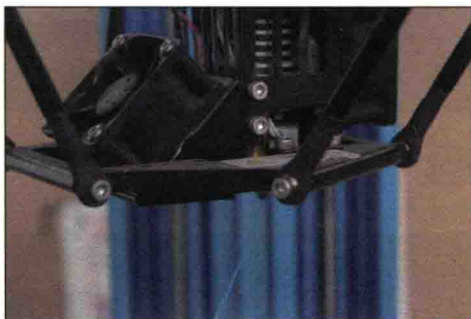
① 填充耗材前，请将其末端适当削尖，以便能顺利插入挤出机构送料齿轮。持续将材料送入导管，直到感觉到送料齿轮将其卡住并拉动。将料盘放置在打印机顶端转盘上。



② 材料填充完成后，在菜单中选择“READY”，打印机会将材料送入喷嘴。



③ 喷嘴能顺利吐丝，表明上料成功。



④ 随后即可选择“READY”进入材料参数设定页面。如果在打印过程中暂停打印并更换材料，需注意可能产生的废料，以免影响打印效果。



⑤ 在最后一步中选择使用的材料（PLA 还是 ABS），完成后选择“OK”返回主菜单。



1.3.3 退料

① 在主界面中通过“↑”和“↓”切换至“Reload”，选中进入换料步骤。



- 2 等待喷嘴被加热至工作温度。



- 3 喷嘴被加热后，若要移除材料，可选择“CONTINUE”；若料管中没有装载材料，可选择“SKIP”跳过此步骤。



- 4 打印机在自动退料时会出现左下图所示进度条。材料的最后一段需要手动从料管末端拉出。



1.3.4 打印

- 1 在开始打印作业前，请确认打印平板上清洁、无异物。打印平板可以使用湿布来擦拭。使用带有可加热铝制平台的 OverLord Pro 打印机时，使用前可在平台上均匀涂抹 3 层固体胶层。为了方便将完成的打印件取下，请在打印前确认胶层已干燥凝固。你也可以通过选择菜单中的“Temperature” → “Heat up Build plate-40°C”来加速干燥过程。



- 2 将存有 Gcode 格式文件的 SD 卡插入位于机器底部的 SD 卡插槽中，插槽右侧会亮起蓝色指示灯，提示 SD 卡被正确插入。
- 3 选择“PRINT”进入到打印模型文件选择界面。选择“Yes”以继续上次打印作业；选择“No”跳至“SD CARD”页面选择新的模型文件。选择“OK”来启动打印作业。

④ 在喷头加热至工作温度后，OverLord 打印机会自动开始打印作业。在打印开始时，指示屏下方会出现提示打印进度和预计完成时间的进度条。打印机在工作时可随时选择“Pause”来暂停打印。暂停打印作业时，打印机会立即保存当前打印进度并在屏幕中显示“Waiting”字样直至进度保存完毕。此进度不会因关闭打印机而丢失，并会在下次启动打印机时继续打印。

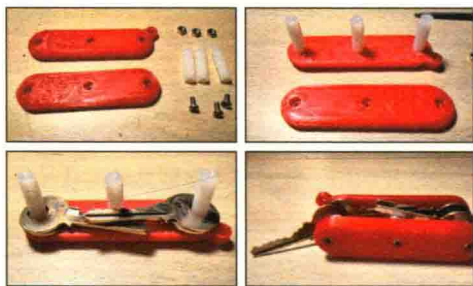


- ⑤ 在打印时，我们可以选择“TUNE”按钮对打印的相关参数进行设置。
- ⑥ 打印完成之后，喷嘴会返回原位。你可使用铲刀将模型铲下来。至此，你应该对 3D 打印的完整流程有

了初步的印象，后面我们将利用 3D 打印机和 Arduino 一起制作电子制作作品。

1.4 思路拓展

其实钥匙扣虽然简单有限，但我们的创意是无限的。你完全可以在原有的基础上进行改进、再创作，或者将自己的创意发挥出来。例如下面的仿瑞士军刀钥匙扣，就很简单又有创意吧？



02

魔戒与台灯

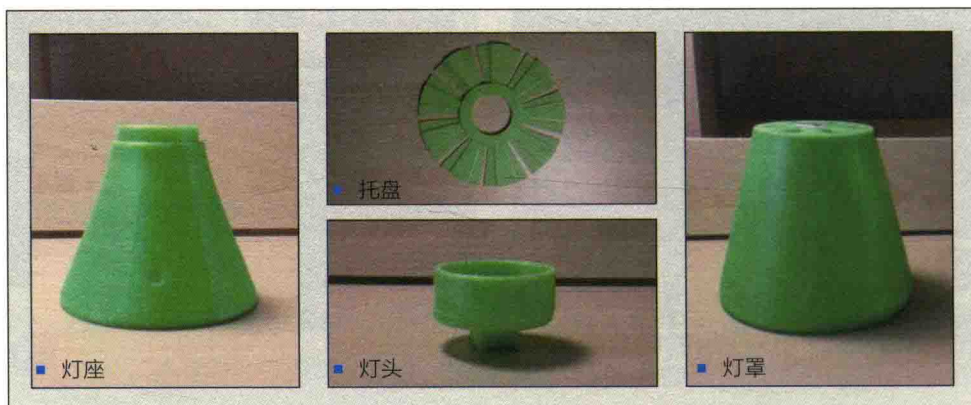


相信大家对电影《指环王》一定不陌生，魔戒具有神秘的力量。不过今天我们用的“魔戒”的作用是点亮一盏3D打印的台灯，效果如题图所示。

我使用 DreamMaker 3D 打印机打印了一枚戒指和一盏台灯（模型文件可从《无

线电》杂志网站 www.radio.com.cn 下载），利用霍尔传感器作为台灯的控制开关。当带有强磁铁的戒指靠近时，台灯被点亮。

台灯由灯座、托盘、灯头、灯罩四部分构成（见图 2.1），组装时按由下向上的顺序依次安装，组合成型。



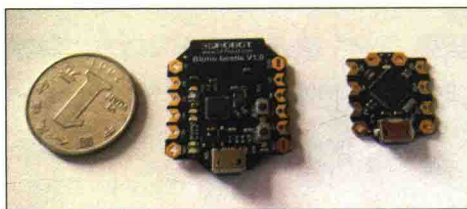
■ 图 2.1 3D 打印台灯零部件

2.1 作品器材

1. Beetle 控制器

Beetle 是 DFRobot 专门为电子制作爱好者和工作坊设计的微型 Arduino Leonardo 控制器。如果你想制作一个小巧、美观的互动作品，那么 Beetle 能助你一臂

之力，因为它只有硬币大小（见图 2.2）！



■ 图 2.2 Beetle 控制器

2. 炫彩 WS2812 LED 灯带

该灯带由 IC、LED 组成。控制器通过控制 IC，对 FPCB 里面的电路进行控制，从而控制 LED 灯条变化不同的效果，如闪烁、追逐、跳变、流水等。

3. 霍尔传感器（Arduino 兼容）

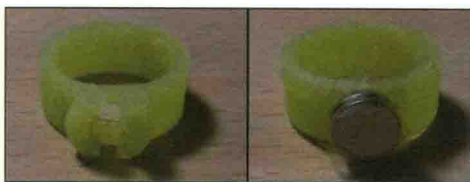
该传感器在无磁场的环境下为断开状态，当磁力足够强时，能够让其导通。整个导通的过程非常快，因而使其成为高效、可靠的开关元件。

4. 强磁铁

5. 高品质公母头跳线（21cm × 30 根）

2.2 制作步骤

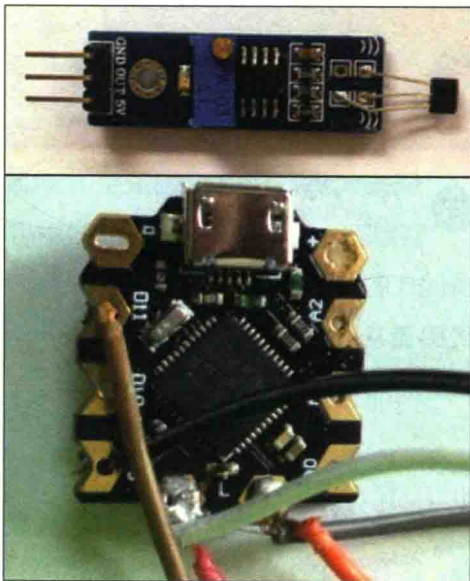
1 用 3D 打印机打印出戒指和台灯零部件：对打印好的戒指适当修模，并将强磁铁安装好。我是用透明胶带把磁铁与戒指绑在一起的。



2 灯带部分：从灯带盘中截取适当长度的灯带，并为截取的灯带焊接好 +5V、D、GND 三根引线。在灯带的安装方面我偷了个懒，直接将灯带绕在了灯头上。建议各位朋友如果自己做的話，在这里可以花点心思好好设计一下。



3 控制板部分的焊接：因为 Beetle 个头较小，所以焊接时应尽量小心，若直接焊接感觉不牢固，可以用点松香。此次用了 Beetle 的数字口 9 和 11，数字口 9 用于连接霍尔传感器的 OUT，数字口 11 用于连接灯带的 D1 口。另外用两根线分别接 VCC 和 GND。



- ④ 组装成品灯：将霍尔传感器安放在 3D 打印的灯底座内，并将传感器头部通过底座上的小孔穿出。座上依次分别叠放上托盘、灯头、灯罩，之后便可组装成下图所示的完整灯体。



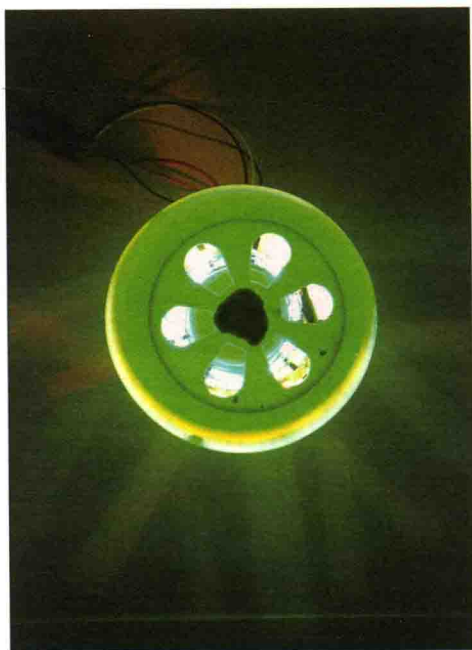
- ⑤ 烧录代码：将代码烧录至 Beetle 控制器，注意在 Arduino IDE 中将 Board 的类型应选择为 Leonardo。代码请从《无线电》杂志网站 www.radio.com.cn 下载。

2.3 作品测试

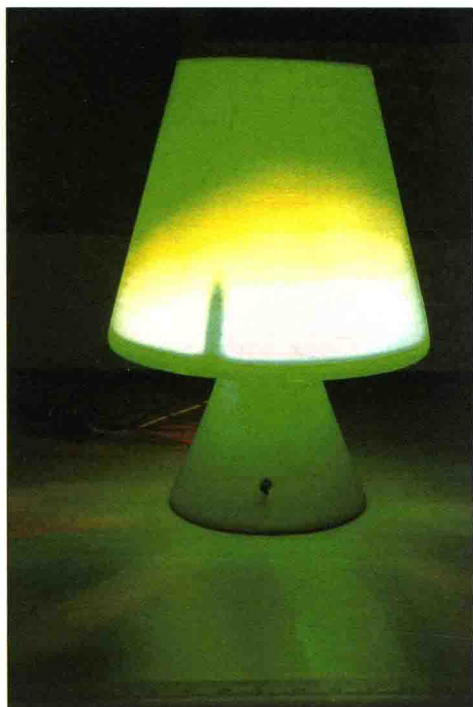
将带有强磁铁的戒指靠近霍尔传感器，看看你的灯亮起来没有（见图 2.3~图 2.5）。



■ 图 2.3 将带有强磁铁的戒指靠近霍尔传感器，灯会亮起来



■ 图 2.4 灯亮了起来



■ 图 2.5 夜晚效果更佳

这个具有简单交互功能的小台灯使用了完整的 3D 打印件作为机械结构（戒指、台灯），电子控制电路部分采用了微型化的 Beetle 结合霍尔传感器制作，实现了通过戒指开灯的一种“魔幻”效果。

你还可以将 Beetle 换成 Bluno Beetle，将强磁铁换成 Bluno Nano+ 按钮，从而实现通过蓝牙控制台灯的效果。有兴趣的朋友可以来玩哦。

03

3D 打印智能自浇花系统

生活中，我们可能会种些花花草草来美化家居环境，但常常遇到这样的结果：一盆倾注了你很多时间、情感和金钱的盆栽，在你外出几天回来后发现它已经枯萎了。不过现在你可以有这样的选择：ParallelGoods 公司发布的 3D 打印的自浇水花盆。该自浇水花盆的设计是自下而上的（见图 3.1、图 3.2），有助于土壤留住养分，从而为你的室内园艺创造一个更为干净、健康的环境。制作所需的器材见表 3.1。



■ 图 3.1 自浇水花盆上部



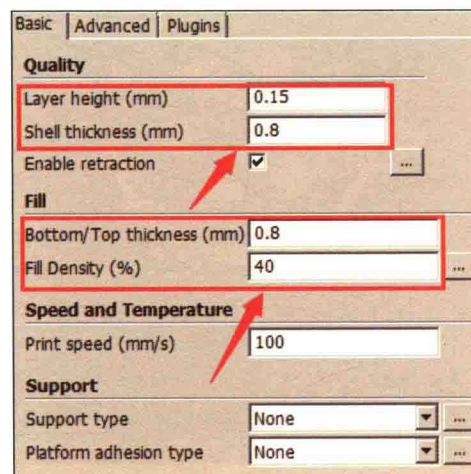
■ 图 3.2 自浇水花盆整体

表 3.1 项目器材

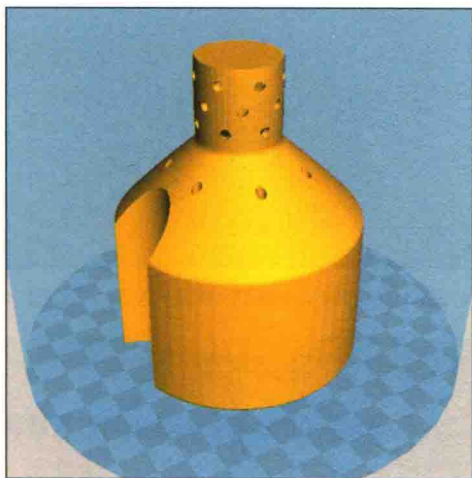
Overlord Pro 3D 打印机
PLA 打印耗材（直径 1.75mm，粉色）
Arduino Uno 控制板
电机扩展板
小水泵
电容式土壤湿度传感器
电池
1kΩ 电阻
I/O 扩展板

这种装置的设计和使用方法都很简单，你只需在上层的盆中填好土，并种上自己喜欢的植物即可，盆子底部的网眼可以让植物的根部获取自己所需要的水分。

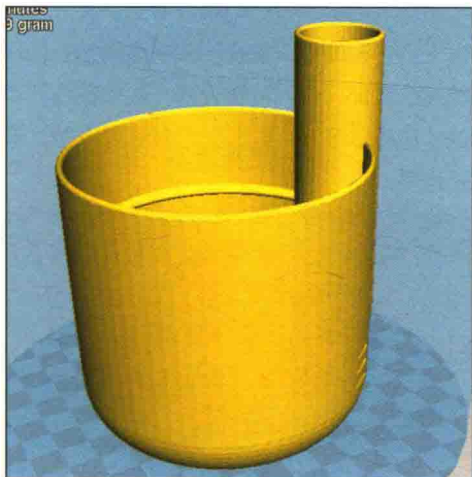
考虑到花盆的坚固性、水密性，我们对打印模型文件的设置如图 3.3 所示，3D 打印模型如图 3.4、图 3.5 所示。



■ 图 3.3 打印设置



■ 图 3.4 花盆上部 3D 打印模型



■ 图 3.5 花盆下部 3D 打印模型

花盆中水被消耗的速度会根据植物品种的不同而有所差别，但是这种自动浇水花盆一次存储的水量能够满足一种典型的草本植物两三次浇水所需要的量。

看到这里你可能会说：“就这么简单？”其实也就这么简单。不过我们不能让它这么简单，毕竟花盆的水量是有限的，如果长期外出，这东西还是不靠谱。于是对此花盆进行功能添加：进行土壤湿度检测，当土壤湿

度小于设定的临界值时，启动电机，灌水到下层水箱。当水位高度到达下层水箱临界高度时，停止灌水，不能让水漫出来。

3.1 电容式土壤湿度传感器校准

在正式检测土壤湿度前，需要校准电容式土壤湿度传感器。烧入下列校准代码至 Arduino 控制板，然后打开串口监视器（注：这里土壤湿度传感器的正极接 Arduino 控制板的 3.3V 引脚），分别读取传感器在空气中和水中的数值来限定一个测量范围。

```
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  int val;
  val = analogRead(2);
  Serial.print(val);
  Serial.println();
  delay(1000);
}
```

打开串口监视器，将波特率根据程序设置为 9600。首先将该传感器放置在空气中读取模拟值，代表干燥时的读数。然后拿一杯水，把传感器插入水中一定深度（做一下标记，此深度为你将要插入泥土的深度），一定不能超过红色警戒线，并记录此时读到的模拟值，代表 100% 湿度。输出数据与湿度成反比，在水中的输出量最小。插入深度如图 3.6 所示。

由于传感器数值会受到入土深度、土壤松紧度的影响，所以只能检测到土壤的相对湿度。我们把湿度分为 3 个等级，分别表示干燥、湿润、非常潮湿。之前记入的两个数据为湿度区间。例如：空气中读数

为 520, 水中读数为 260, 这样就可以分为 (520,430]、(430,350]、(350,260] 这 3 个区间, 分别代表干燥、湿润、非常潮湿。



■ 图 3.6 校准时传感器推荐插入的土壤深度

注意：由于此传感器依据电容感应原理来监测土壤湿度，所以土壤湿度不同、松紧度不同、插入深度不同，会体现出不同的读数。即使在同一个地方、同样的深度，在第二次插入时，由于第一次拔出时已经造成了泥土的松动，可能读到的湿度也会比第一次低。

3.2 电路连接

3.2.1 传感器连线

将 I/O 扩展板、电机扩展板、Arduino Uno 按对应 DVA 口叠加（见图 3.7），土壤湿度传感器接模拟口 2（注意 3 根线分别接在 I/O 扩展板的 A2、+5V、GND 上）。



■ 图 3.7 各类板卡叠加

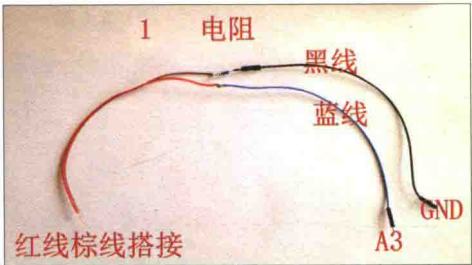
3.2.2 信号下拉

下层水箱里的水是否到达临界位置是通过两根导线是否连通来判断的。这里使用了 1kΩ 电阻做信号下拉，避免干扰和噪声，如果没有电阻，模拟口就会悬空，悬空就会有干扰。将信号下拉线一头的金属部分粘在花盆的底部，临界位置导线一端粘在临界点处（见图 3.8）。当水位达到临界点时，电路连通。

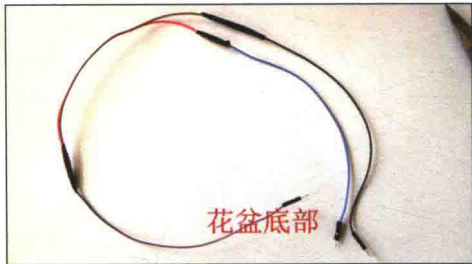


■ 图 3.8 底部和临界位置导线安装

信号下拉线的做法：取杜邦线若干，1kΩ 电阻两端分别与黑色、棕色线焊接，蓝色线与红色线焊接，红色线和棕色线焊接（见图 3.9）。黑线一端接在 I/O 扩展板 GND，蓝色线一端接 I/O 扩展板 A3，红色和棕色线焊接后与紫色线搭接，紫色线最后粘在下层的底部。这样就完成一个信号下拉线。最后将各焊接点用黑色绝缘胶布粘牢，避免相互搭线（见图 3.10）。



■ 图 3.9 信号下拉线做法



■ 图 3.10 信号下拉线焊点绝缘

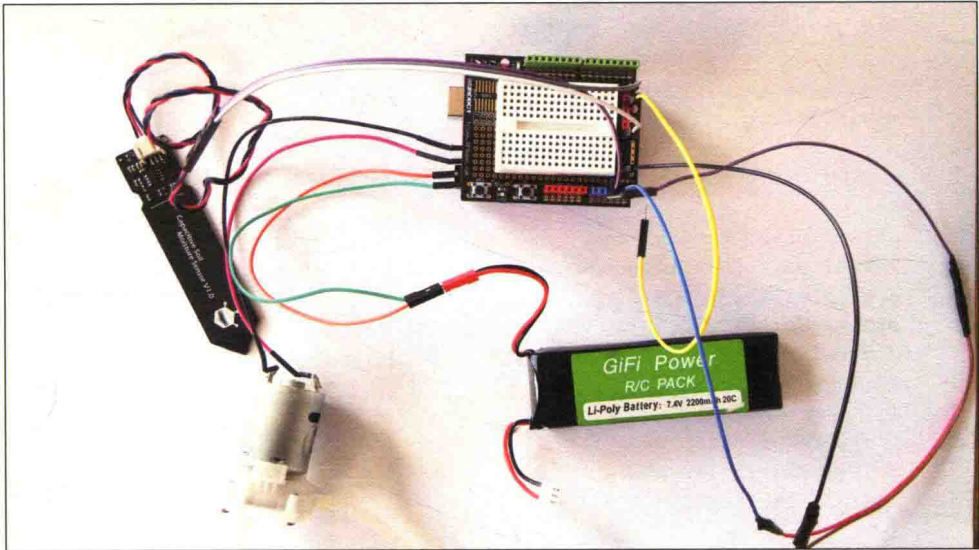
3.2.3 水泵连线

从下表中我们可以看出驱动电机需要有 3 个参数：首先就是方向，对应 I/O 口的高

电平为一个方向，低电平为反方向；然后就是速度，这里使用 PWM 的方式控制，给予其不同的占空比会获得对应的速度；最后是刹车，刹车的含义为是否将电机的两极短接，若短接两极，电机旋转会有非常大的阻尼，能量由续流二极管吸收，进而起到制动的作用，因此，要想电机转动，这个 I/O 口应该是低电平。知道了原理，驱动电机应该就很简单了吧？注意：使用时应保证电机驱动板与 Arduino 控制板电源完全独立，使用两块电池或者两个独立的电源，保证其电气隔离性。

	功能	电机 A	电机 B
1	方向	D12	D13
2	速度	D3	D11
3	制动	D9	D8

将水泵的红色线（正极）接入 A+ 接线柱，黑色线（负极）接入 A-接线柱。外接电源的正极接 VIA，负极接 GND。总体连线图 3.11 所示。



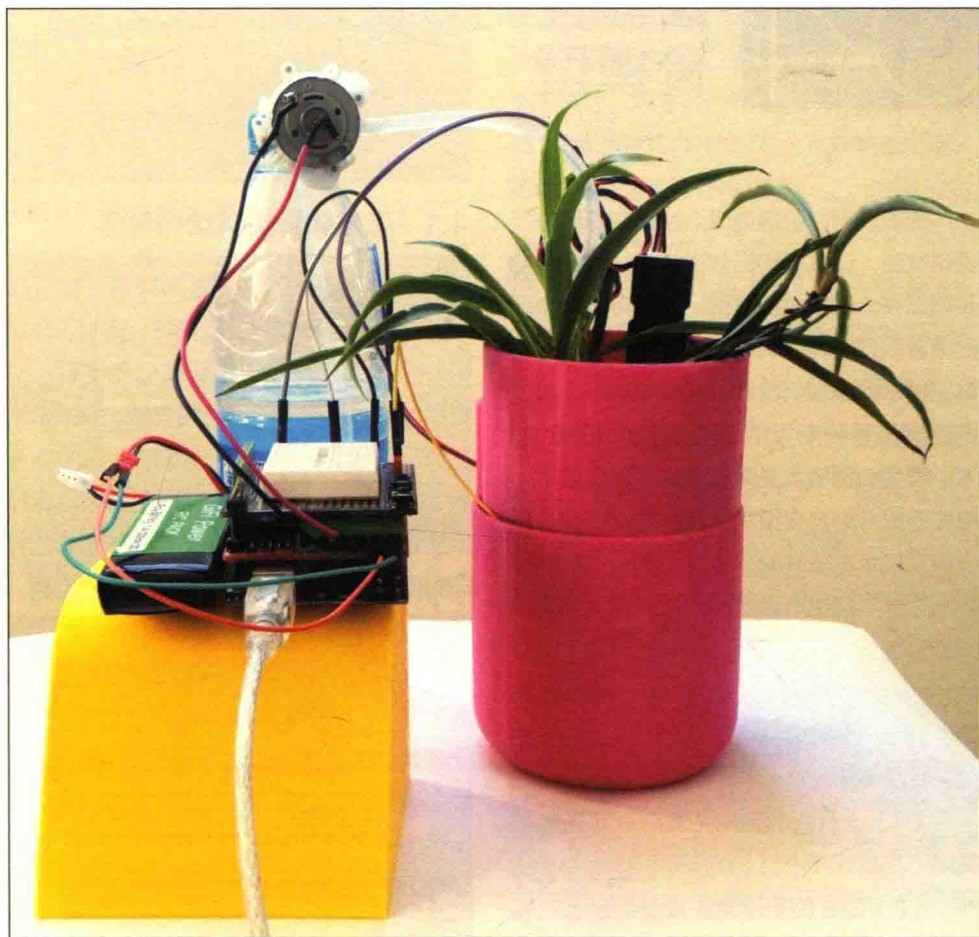
■ 图 3.11 总体连线图

3.3 程序代码

```
const int AirValue =900;
in the air;
const int WaterValue = 300;
in the water;
int intervals = (AirValue - WaterValue)/3;
int soilMoistureValue = 0;
void setup()
{
    Serial.begin(9600);
    pinMode(12,OUTPUT);
    pinMode(3,OUTPUT);
    pinMode(9,OUTPUT);
}
void loop()
{
    soilMoistureValue = analogRead(A2);
    int n=analogRead(A3);
    if(soilMoistureValue > WaterValue && soilMoistureValue < (WaterValue + intervals))
    {
        Serial.println("Very Wet");
        digitalWrite(9, HIGH);
        digitalWrite(3, LOW);
        digitalWrite(12, LOW);
    }
    else if(soilMoistureValue > (WaterValue + intervals) &&
soilMoistureValue < (AirValue - intervals))
    {
        Serial.println("Wet");
        digitalWrite(9, HIGH);
        digitalWrite(3, LOW);
        digitalWrite(12, LOW);
    }
    else if(soilMoistureValue < AirValue && soilMoistureValue > (AirValue - intervals))
    {
        Serial.println("Dry");
        Serial.println(analogRead(A3));
        digitalWrite(9, LOW);
        digitalWrite(3, HIGH);
        digitalWrite(12, HIGH);
        if (n>1)
        {
            digitalWrite(9, HIGH);
            digitalWrite(3, LOW);
            digitalWrite(12, LOW);
        }
    }
    delay(1000);
}
```

将上述代码烧录到 Arduino 控制板中，测试一下你的浇花系统（见图 3.12）吧。

最后别忘记添加储备水箱，你的花多长时间浇水，就看你的水箱大小了。



■ 图 3.12 最终效果图

04

语音控制 3D 打印恐龙头骨模型

语音识别，说得通俗点就是与机器进行语音交流，让机器明白你说什么，这是人们长期以来梦寐以求的事情。我们可以形象地把语音识别比作“机器的听觉系统”。语音识别技术就是让机器通过识别和理解过程，把语音信号转变为相应的文本或命令的技术。语音识别技术主要包括特征提取技术、模式匹配准则及模型训练技术三个方面。语音识别技术在车联网、手机上都有广范的应用。你是不是也想做出点什么来呢？今天我们就带着大家一起使用 DFRobot 的 Voice Recognition 语音识别扩展板（Arduino 兼容）制作一个可以用语音控制的桌面恐龙头骨。其具体功能如下。

（1）语音控制恐龙头骨的眼睛亮起和关闭（左右分开控制）。

（2）语音控制恐龙头骨在云台上左转和右转。

（3）语音控制恐龙头骨发出叫声。

制作所需的器材见表 4.1。

表 4.1 项目器材

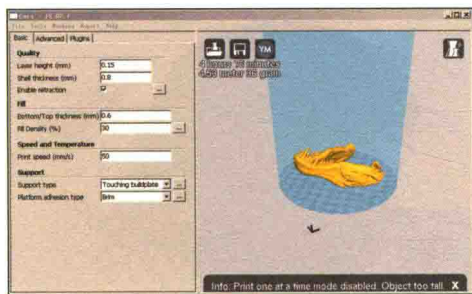
1.75mm PLA 3D 打印耗材（橙色，1kg）
Arduino UNO
DFVoice Recognition 语音识别扩展板（Arduino 兼容）
DF15RSMG 360° 舵机 19kg
全方位蜂巢云台 LED（蓝色）
DFPlayer Mini 播放器模块与小扬声器

4.1 恐龙头骨机械结构制作

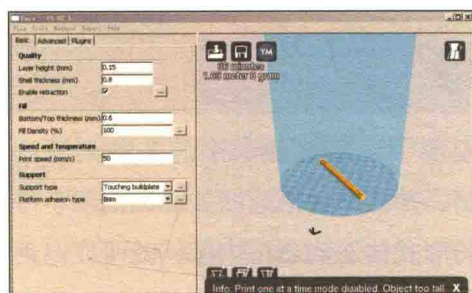
本次的 3D 打印机器人共由 4 部分组成：头盖骨、下颚骨、支撑柱、底板（见图 4.1~图 4.4）。考虑到系统的稳定性，对打印的部件设置了不同的填充率：头盖部为 30%，下颚骨为 30%，支撑柱为 60%，底板为 60%，这样在转动过程中就可以保持系统稳定。另外，由于头盖骨和下颚骨的打印时间相对较长，为了保证打印顺利完成，我们对其进行了特殊设置：“supprot type: touching buildplate”选项是给需要的地方加支撑；“platform adhesion type:brim”选项是给底边加裙边，防止打印过程中底面翘边。打印完成后，将上述 4 部分组合成型（见图 4.5）。



图 4.1 头盖骨 3D 模型



■ 图 4.2 下颚骨 3D 模型



■ 图 4.3 支撑柱 3D 模型



■ 图 4.4 底板 3D 模型

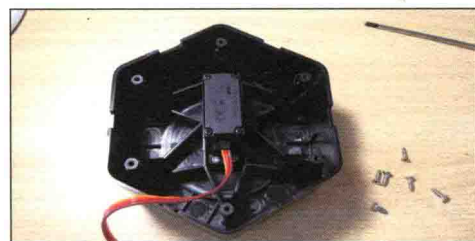


■ 图 4.5 打印完成的恐龙头骨模型

4.2 云台部分组装

这次我们使用了 DF 的蜂巢云台，此云台外壳采用高强度 ABS 塑料制作，其六边外形不仅显得简洁、美观，而且结合底盘的搭扣设计，可与云台相互衔接，组成蜂巢阵列。我们将打印好的恐龙头骨安装在上面，就可以实现展示平台的功能。

① 将舵机摆放在云台外壳内侧相应位置，孔位对齐，使用 M4×10mm 圆头十字自攻螺丝将舵机固定住。注意拧螺丝时，将螺丝拧进孔位后，不要直接拧紧，先拧进其他螺丝，待 4 个螺丝都拧进孔位后，轮流旋入，并均匀拧紧，切勿上一个紧一个。



② 将舵机电线从外壳底部边缘槽位引出，电线槽位与其他槽位略有不同，不会搞错，这也反映了 DFRobot 在做产品的细心之处。将底板摆放在云台外壳底部相应位置，底板边缘凸起结构和外壳底部边缘凹槽匹配，卡入槽位。如槽位不匹配，请转换一个槽位继续安装。使用 M3×10mm 沉头十字自攻螺钉将底

板固定住，螺丝拧紧后，应该陷入底板 0.5mm 左右。



- 3** 将塑料舵盘固定在舵机上，然后依次将滚针轴承放于上方。



- 4** 将云台转盘安装到舵盘上，注意对齐孔位，拧入螺丝。



4.3 电路连线

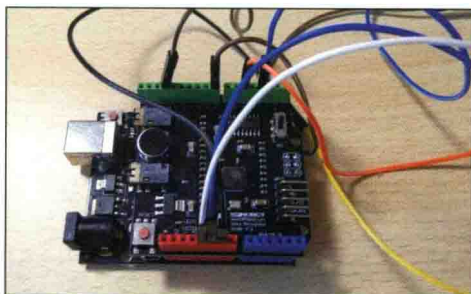
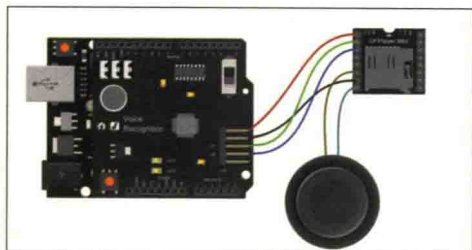
- 1** Voice Recognition 语音识别模块是一款只需要在主控 MCU 的程序中设定好要识别的关键词语列表，并动态地把这些关键词语以字符的形式传送到芯片内部，就可以对用户说出的关键词语进行识别，不需要用户事先训练和录音的非特定人语音识别模块。该模块可以设置 50 项候选识别句，每个识别句可以是单字、词组或短句，长度为不超过 10 个汉字或者 79 个字母的拼音串，可由一个系统支持多种场景。将 Voice Recognition 语音识别扩展板对应插在 Arduino UNO 上。注意在使用 MP3 模块时，请将通信开关拨置到 UART 端。模块与 Arduino 引脚的连接为：语音模块连接 D2、D4、D9、D10、D11、D12、D13，MP3 模块连接 D0、D1、A4、A5。

- ② 对 LED 连线进行延长，并将 LED 固定在恐龙头骨眼眶处。两颗蓝色 LED 分别接 Arduino 控制板上的数字口 D3、D8。

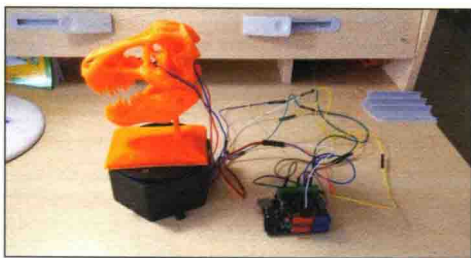


- ③ 云台舵机的 3 根线（橙色、红色、棕色）分别与 Arduino 的数字口 D5、VCC、GND 连接。

- ④ MP3 Mini 播放模块和小扬声器与语音识别模块的连线如下图所示，由于 MP3 模块使用了数字口 D0、D1，所以要在烧录完程序后再连线，否则可能会出现程序烧录失败的情况。文件夹需要命名为“mp3”，放置在 SD 卡根目录下，而 MP3 文件需要命名为 4 位数字，例如“0001.mp3”，放置在 MP3 文件夹下。如果需要中英文命名，可以添加在数字后面，例如“0001hello.mp3”或者“0001 后来.mp3”。



- ⑤ 将恐龙底板固定在云台上。



4.4 代码编写

本例中使用了库文件 voiceRecognition 和 DFPlayer-Mini-mp3，要将这两个文件夹复制到 Arduino\libraries 文件夹下，否则在程序编译过程中会报错。

```
#include <avr/wdt.h>
#include <Servo.h>
#include <VoiceRecognition.h>
VoiceRecognition Voice;
#define leftLed 8 // 定义 LED 引脚
#define rightLed 3 // 定义 LED 引脚
#include <SoftwareSerial.h>
#include <DFPlayer_Mini_Mp3.h>
bytesensorPin = 2;
byte indicator = 13;
Servo myservo;
intpos = 0;
void setup()
{
    mp3_set_serial (Serial);
    //set Serial for DFPlayer-mini MP3 module
    mp3_set_volume (30);
    myservo.attach(5);
    Serial.begin(9600);
    pinMode(leftLed,OUTPUT);
    // 初始化 LED 引脚为输出模式
    pinMode(rightLed,OUTPUT);
    // 初始化 LED 引脚为输出模式
    digitalWrite(leftLed,LOW);
    //LED 引脚低电平
    digitalWrite(rightLed,LOW);
    //LED 引脚低电平
    //mp3_set_serial (Serial);
    // 设置 MP3 模块通信方式
    //mp3_set_volume (15); // 音量调节
    Voice.init();
    // 初始化 VoiceRecognition 模块
    Voice.addCommand("zhuo deng",0);
    // 添加指令, 参数 1 为指令内容, 参数 2 为指令标签 (可重复)
    Voice.addCommand("you deng",1);
    Voice.addCommand("guan deng",2);
    Voice.addCommand("前转",3);
    Voice.addCommand("后转",4);
    Voice.addCommand("bo fang",5);
    Voice.addCommand("ting zhi",6);
    Voice.start();// 开始识别
    wdt_enable(WDTO_1S);// 打开看门狗 (防止死机)
}
void loop()
{
    switch(Voice.read())
    // 判断识别内容, 在有识别结果的情况下 Voice.Read() 会返回该指令标签, 否则返回 -1
    {
        case 0:// 若是指令 "LEFT DENG"
```

```

digitalWrite(leftLed,HIGH);
// 点亮 LED
Serial.println("received'LEFTdeng',command flag'0'");
// 串口发送 received"LEFTdeng", command flag"0"
break;
case 1:// 若是指令 "rightdeng"
digitalWrite(rightLed,HIGH);
// 熄灭 LED
Serial.println("received'rightdeng',command flag'1'");
// 串口发送指令
break;
case 2:// 若是指令 "guan deng"
digitalWrite(rightLed,LOW); // 熄灭 LED
digitalWrite(leftLed,LOW); // 熄灭 LED
Serial.println("received'guandeng',command flag'2'");
break;
case 3:// 若是指令 "qianzhuan"
for(pos = 0; pos< 180; pos += 1){
    myservo.write(pos); // 给舵机写入角度
    delay(15); }
Serial.println("received'qianzhuan',command flag'3'");
break;
case 4:// 若是指令 "houzhuan"
for(pos = 180; pos>=1; pos-=1) {
    myservo.write(pos); // 写角度到舵机
    delay(15); }
Serial.println("received'houzhuan',command flag'4'");
break;
case 5:// 若是指令 "bo fang"
mp3_play (0001);
Serial.println("received'bo fang', command flag'5'");
break;
case 6:// 若是指令 "ting zhi"
mp3_stop ();
Serial.println("received'ting zhi', command flag'6'");
break;
}
wdt_reset();
}

```

4.5 作品测试

完成以上步骤后，大家可以测试一下作品的效果，分别对语音识别模块说出“左灯”“右灯”“关灯”“前转”“后

转”“播放”“停止”语音指令，看看恐龙头骨的左右眼和云台、MP3 语音播放是否正常运作。

05

3D 打印膝关节辅助装置

动力外骨骼最早是出现在科幻小说中的机械及电子设备，是一种能够增强人体能力的可穿戴机器（见图 5.1）。它能够帮助人们跑得更快、跳得更高、携带更多重物，并可帮助穿戴它的人在战场、建筑工地或者其他有危险的地方生存下来。这项技术已经有 20 多年的历史，但是出于技术原因，现在还没有广泛应用，只存在于实验室或者试验场中。



■ 图 5.1 动力外骨骼

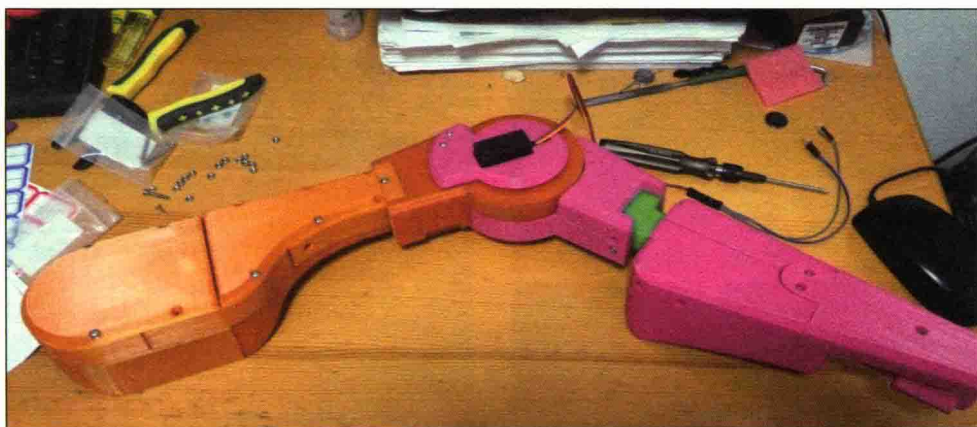
不过，来自日本的“动力外骨骼达尔文”（见图 5.2、图 5.3）利用 3D 打印技术，让你在家里就可以自己打印人体外骨骼。下面就给大家介绍针对膝关节骨性关节炎患者的膝关节辅助外骨骼。制作所需的器材见表 5.1。

表 5.1 项目器材

3D 打印机（笔者用的是 Overlord Pro）
PLA 打印耗材（金色、粉红、绿色）
DF15RMG 机器人舵机
32 路舵机控制板
SONY PS2 手柄与接收器
多媒体集线器
缠绕环
舵机延长线
开关
M3×10、M3×20、M3×30、M3×50、M3×100 等扁头内六角螺丝，10 套



■ 图 5.2 膝关节辅助外骨骼



■ 图 5.3 最终作品

5.1 工作原理

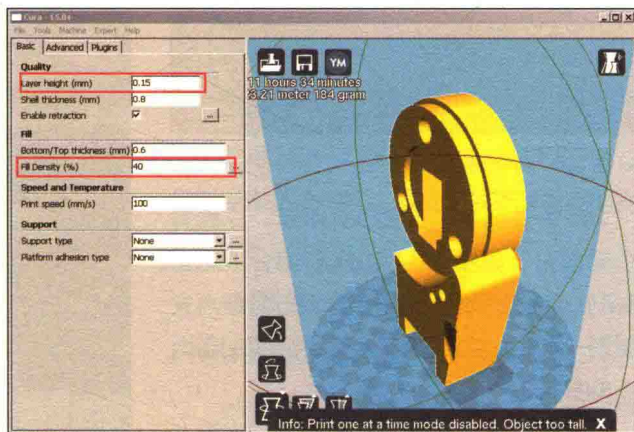
动力外骨骼通过传感器来收集使用者的活动信息，这些信息传递给信息处理器进行处理，然后启动相应的机械部件来输出能量。在动力外骨骼中可以使用的传感器种类很多，包括角辨向器、肌电传感器、地面

传感器、肌肉压力传感器等。同样，用于输出的机械系统也有很多选择。这里选择的是 PS2 手柄来控制舵机转动，从而产生对膝关节的辅助动力。

5.2 制作机械部分

5.2.1 3D 打印

从网站下载 3D 打印文件 (<http://www.thingiverse.com/thing:515259/zip>), 用 Cura 进行切片处理, 这里考虑到关节承受动力, 将打印填充率设置为 40%, 层高设置为 0.15 (见图 5.4), 然后开始打印。其他部件的打印设置与之相似。



■ 图 5.4 打印设置

5.2.2 部件组装

由于本作品中 3D 打印部件较多，我们将其分成 3 部分（控制器部分、膝关节部分、小腿关节部分），从上到下进行组装。

控制器部分如图 5.5 所示，用于安置舵机控制板、电池等电子器件。



■ 图 5.5 控制器部分

膝关节部分可按图 5.6 所示，主要有主动部分（见图 5.6）和从动部分（见图 5.7），这里需要在这两部分进行打磨再安装到一起，以便减小转动过程中所受的阻力（见图 5.8）。

小腿部分可按图 5.9 所示进行安装，这里的部件设计成可根据人体小腿长度适当

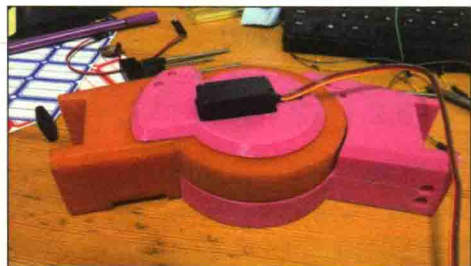
调整长度的样式。然后如图 5.10 所示把控制器部分、膝关节部分、小腿部分组合到一起。



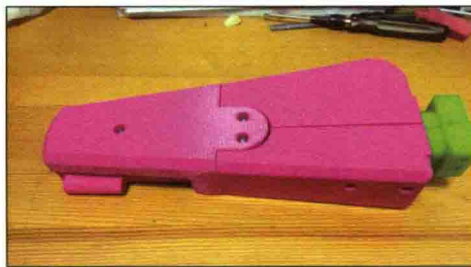
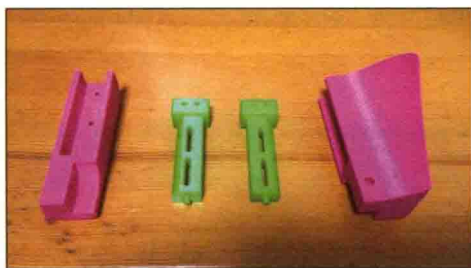
■ 图 5.6 将舵机放入主动部分中，并用螺丝固定



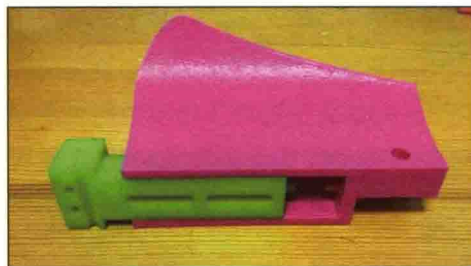
■ 图 5.7 从动部分



■ 图 5.8 用螺丝固定主动部分和从动部分，并将固定板安装到舵机上



■ 图 5.9 可适当调整长度的部件



■ 图 5.10 把控制器部分、膝关节部分、小腿部分组合到一起



5.3 电子部分

5.3.1 安装驱动

使用 Mini-USB 线连接舵机控制板，根据操作系统不同，选择安装不同版本的驱动程序。

5.3.2 安装 Microsoft .Net Frameworks 2.0

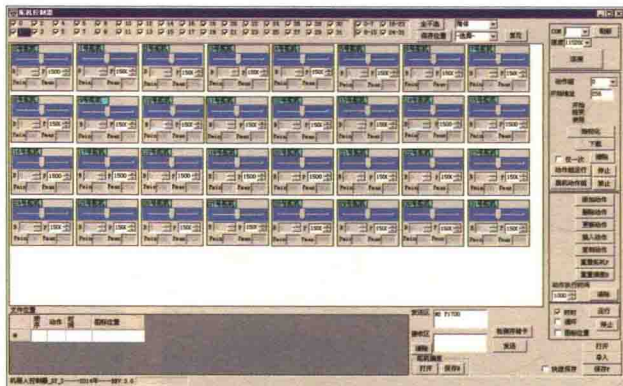
如果已经安装了 .Net Frameworks 或者有更高版本，可跳过此步。如果不确定是否已经安装，可尝试打开上位机软件，能打



开则无需安装，如果打不开则要安装。

5.3.3 上位机软件介绍

图 5.11 所示为上位机软件界面。左边为舵机图标操作窗口，可以拖动滑块设置舵机角度；最上方可以选择需要显示的舵机（勾选舵机编号，即可显示对应舵机）；右边为舵机图标位置保存窗口，可以设定一个或几个舵机的动作组，将其保存并下载到控制器中，实现脱机运行。



■ 图 5.11 上位机软件

5.3.4 通信设置及舵机设置

COM 口要根据舵机控制板的实际情况进行选择，默认通信速度为 115200bit/s，特殊情况下可使用 9600bit/s，如图 5.12 所示。



■ 图 5.12 通信设置

舵机口指示条也可随意拖动，B 表示舵机偏差（默认为 0），即舵机的相对位置，范围为 -100~100；P 表示舵机位置（默认为中位 1500），范围为 500~2500。这里只使用了 17 号舵机，设置如图 5.13 所示。



■ 图 5.13 舵机设置

5.3.5 舵机控制板供电、PS2

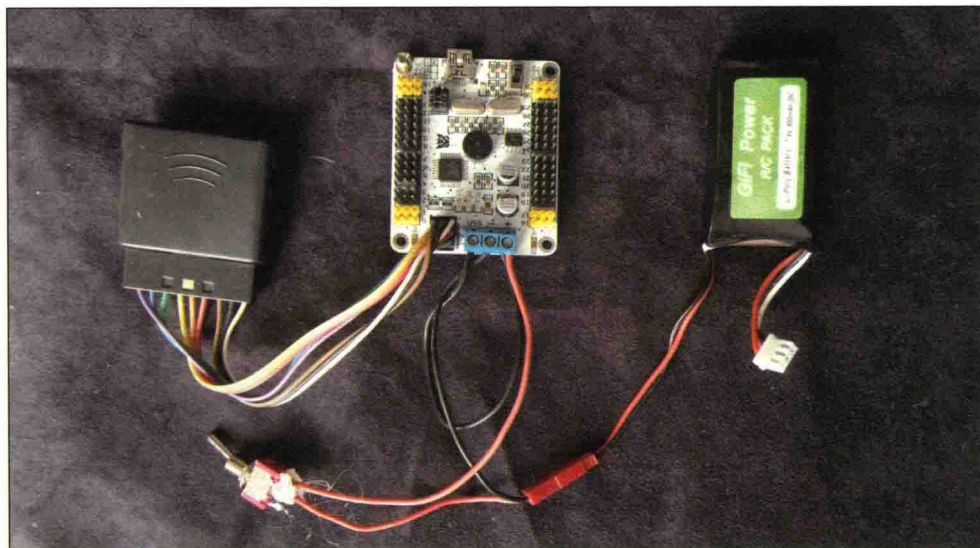
手柄接收器、舵机连接方案

首先要确定自己所使用的舵机的供电电压（一般为 5~7.2V），主板供电电压 VSS 为 7~12V。以 QSC32E 舵机控制板为例，接线图如图 5.14 所示，此方案对于 7.4V 电压的机器人舵机且舵机数量较多的情况，是最稳定、最理想的。

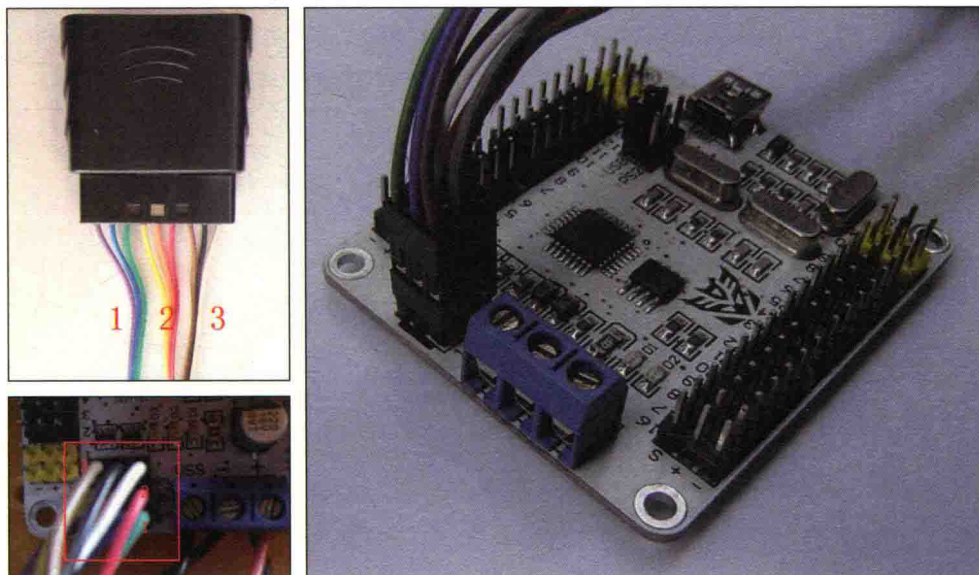
注意舵机控制板的 USB/PS2 转换开关要放置到 USB 位置。在舵机口上连接一个舵机，例如接在 17 号舵机口（接这个口便于测试上位机软件，也便于测试摇杆），注意舵机排针接口的 S、+、-与舵机线的对应关系，一般来说白、红、黑三色的线，白色接 S，红色接 +，黑色接 -；如果是黄、红、棕三色的线，则黄色接 S，红色接 +，棕色接 -。只选择 17 号舵机，然后左右拖动 17 号舵机的滑块，这时通信指示灯 D1 会跟着

同步闪动且舵机会跟着左右转，舵机测试完毕。

舵机控制板上的 PS2 手柄接收器接口
如图 5.15 所示。



■ 图 5.14 舵机控制板与电池、PS2 手柄接收器的连接



■ 图 5.15 舵机控制板上的 PS2 手柄接收器接口

5.3.6 联调

将 USB/PS2 转换开关拨到 PS2 挡位上，这里有两种工作模式：动作组模式和 6 自由度单舵机微调模式。默认为动作组模式，按下按键执行动作组，松开按键停止动作组；6 自由度单舵机微调模式用于通过手柄来控制机械臂，最多可以控制 6 个舵机，这里使用这种工作模式。

每按一次 SELECT+START 可切换一次工作模式，若听到清脆的“滴滴”声后，

PS2 手柄接收器上的灯不再闪烁，则表明连接成功（见图 5.16），此时按动手柄上的方块和圆形按钮就可以控制舵机转动了（见图 5.17）。

最后将舵机控制板放置到 3D 打印件内部，再将 3D 打印的膝关节辅助装置固定到腿上，就可以用手柄来测试一下它对行走的帮助有多大了。



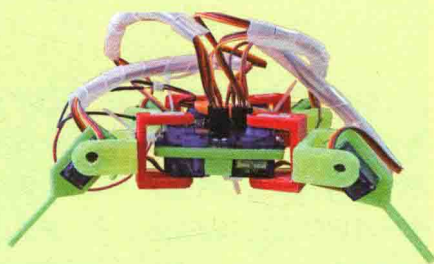
■ 图 5.16 PS2 手柄与接收器连接成功



■ 图 5.17 按动手柄上的方块和圆形按钮就可以控制舵机转动了

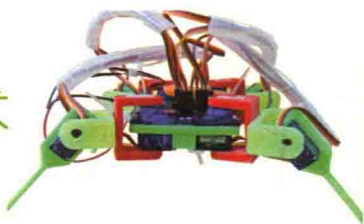
第 2 章

制作个性化机器人



- 06 借助 3D 打印技术制作 4 足机器人
- 07 3D 打印的 4 足机器人脱机版
- 08 用 3D 打印部件制作跳舞机器人 BOB
- 09 让智能手机变身为会走路的机器人

06

借助 3D 打印技术
制作 4 足机器人

使用 3D 打印机打印出自己设计的或者喜欢的个性化机器人，想必是每个创客都想做的事情。今天给大家分享的案例是一个 4 条腿的爬虫机器人，这里只是提供一个可移动的平台，具体功能希望大家继续做下去，拓展出更多有趣的玩法。让我们一起来享受这造物的乐趣吧！制作所需的器材见表 6.1。

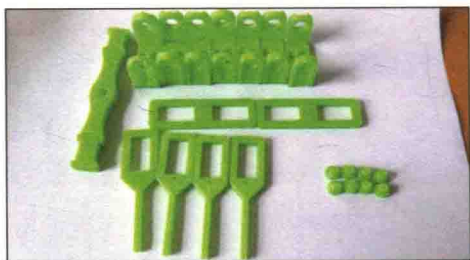
表 6.1 项目器材

序号	名称	数量
1	$\phi 1.75\text{mm}$ PLA 3D 打印耗材(荧光绿、红、黄)	3 kg
2	SG90 舵机	8
3	DFRobot 24 路舵机控制板	1
4	3.7V 锂电池	1
5	3.7V 锂电池充电器	1

6.1 机械结构部分制作步骤

1 用 3D 打印机打印出虫子的机械结构（3D 模型下载自网络）。有小伙伴反映“模型文件打印出的关节部分，舵机很难安装进去”，这里提供 2 种解决方案：一种是对打印后的关节部分进行修模，这是个苦活，需要时间和耐心；另一种是对模型文件

进行优化，适当放大比例。这里友情提示，考虑到 PLA 的弹性和硬度问题，建议打印填充率设置为 100%。本次打印模型的过程比较顺利，打印难度较低，共耗时 9h24min。



2 拿出打印好的腿部，将 1 个舵机安放进去。如果感觉尺寸不合适，可用锉刀适当修理，如下图所示，平整安装好腿部舵机，并用小螺丝固定（注意在安装舵机时，对于舵机的信号线要小心处理，不要弄断）。其他 3 条腿的安装方法与此相同，依此操作即可。



3 拿出打印好的肩部，将 2 个舵机分别安装在肩部左、右固定位中。

这里再次提醒一定要注意舵机信号线，千万别弄坏，否则后面你就玩不了。



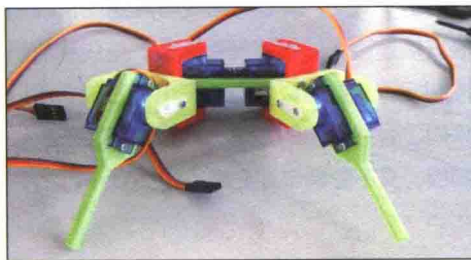
- 4 取出两个关节，用 M3 螺丝固定。
注意两个关节呈 90° 垂直安装，如下图所示。



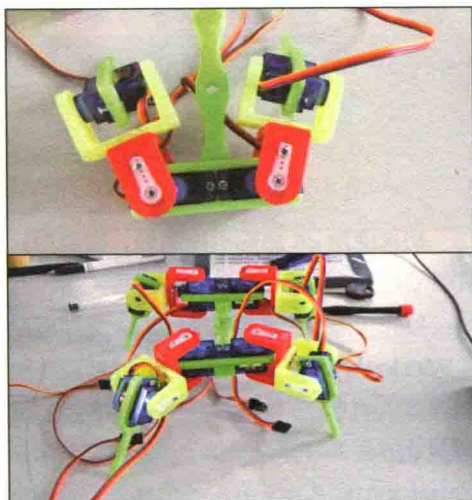
- 5 取出前面步骤中安装好的腿部和关节，将腿部舵机轻轻地塞进关节固定件中，并用螺丝固定舵机，这里建议使用两颗螺丝固定舵机的舵盘，防止舵机在转动过程中松动。



- 6 将前面步骤中安装好的腿部、关节部分与肩部组合在一起。



- 7 将前面步骤中安装好的部分与躯干组合在一起，并用螺丝固定。



- 8 初始化舵机角度，这里的初始化只是一个相对的位置，其目的是让舵机的活动范围尽量在预设的范围内，因为一加电后产生的脉冲将引发舵机的角度改变。我们这里将关键部分尽量向里收，作为起始点。腿部也遵循此规则，向内收缩到极限位置。

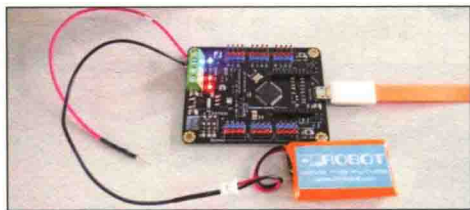
- 9 为了简单起见，在制作时，将关节部分的舵机分别连接在控制板的 0、2、4、6 口，各关节对应的腿分别连接在 1、3、5、7 口。

6.2 电子部分制作步骤

6.2.1 驱动程序的安装

此次使用的控制板是 DFRobot 的威龙 (Veyron) 24 路舵机控制板 (见图 6.1)，其缺点是需要电池供电、不能脱机

运行（选用它时，我以为它可以脱机使用）。



■ 图 6.1 威龙 24 路舵机控制板

初次使用 24 路舵机控制板，需要为其安装 DFU 和虚拟串口两个驱动程序。

首先按照如下步骤安装 DFU 驱动程序（Windows 环境下）：将舵机控制板连接到计算机的 USB 接口，按一下左下角的复位键（印有“RET”字样），这时蓝色指示灯会快速闪烁 6 次，然后慢闪烁几次。再次按复位键，这次在蓝色指示灯快速闪烁 6 次期间按下另一个键（右上角，印有“BOOT0”字样）并保持不放，直到指示灯开始慢闪烁时放开。蓝色指示灯会一直闪烁，使你有机会安装 DFU 驱动程序。Windows 会提示你需要驱动程序，人工指定驱动程序所在目录位置，选择 DFservo 文件夹中的 mapleDrv\dfu。

安装虚拟串口驱动程序：复位舵机控制板的 RET 按键，等待蓝色指示灯停止闪烁。此时 Windows 会提示安装驱动程序，同样，人工指定驱动程序所在目录为 DFservo 文件夹中的 mapleDrv\serial。

上述过程算是比较顺利的，但是有些小伙伴也可能会遇到如图 6.2 所示的问题。

当用 USB 数据线连接计算机时却得到“INF 中的服务安装段落无效”的错误提示，在设备管理器中看到“USB MODEM”

设备出现一个黄色问号，出现这个问题并不是设备或驱动程序出了问题，而是用户采用了深度简化版的操作系统。只需在他人的电脑中分别找到 C:\windows\inf 目录下的 mdmcpq.inf 文件和 C:\WINDOWS\system32\drivers 下的 usbser.sys 文件，并将它们复制到本机相同的目录，重启计算机，即可解决这个问题。



■ 图 6.2 无法安装硬件

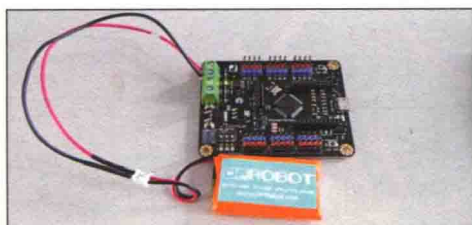
安装好驱动程序，在设备管理器中即可看到相应的设备（见图 6.3）。



■ 图 6.3 在设备管理器中看到相应的设备

6.2.2 电池供电问题的解决

由于 3.7V 锂电池无法与威龙舵机控制板直接连接，如果要剪断接头，连线到舵机控制板，又会造成锂电池充电不方便；所以采用了导线引出的方法，既方便连接舵机控制板，又不影响电池充电（见图 6.4）。



■ 图 6.4 将锂电池电源用导线引出

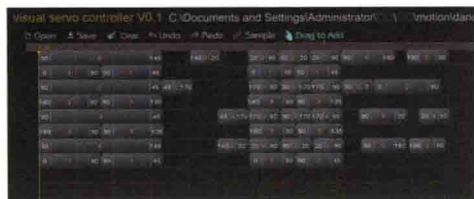
6.2.3 联机调试

联机调试采用的是上位机软件，软件界面如图 6.5 所示，我想这张软件使用说明图足以满足大家的需要了。



■ 图 6.5 上位机软件使用说明

最后提供几种虫子机器人运动方式（前进、跳舞、站立）的截图（见图 6.6），希望小伙伴们能够玩出更多花样。

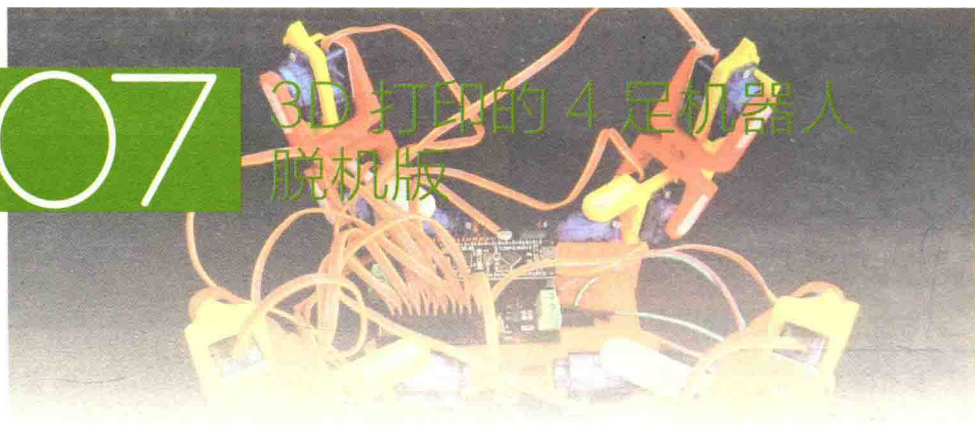


■ 图 6.6 虫子机器人的几种运动方式

6.3 作品总结

这个作品利用 3D 打印制作虫子机器人的机械结构，再通过舵机控制板实现对机器人运动的控制，不过这次制作的机器人是联机版的，要通过 PC 信号控制舵机，不能脱机使用，是个小遗憾。当然，让虫子机器人动起来只是第一步，后面还可以继续拓展，例如在它身上搭载各种传感器，并将数据上传或者返回，从而代替人去侦测环境数据等。

07

3D 打印的 4 足机器人
脱机版

我们已经向大家介绍过一个 3D 打印的 4 足机器人的制作过程，遗憾的是，当时的 4 足机器人不能脱机运行。现在就让我们来弥补这个缺憾，让 4 足机器人甩开电脑，走起！

制作所需的器材见表 7.1。

表 7.1 项目器材

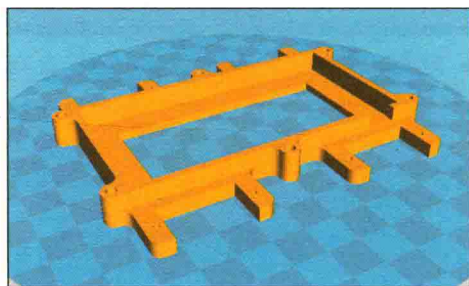
序号	名称	数量
1	Arduino Nano	1
2	Arduino Nano I/O 扩展板	1
3	HX11A 塑料齿 9g 舵机 双轴设计	12
4	9V 电池转换插头	1
5	品胜 9V/250mAh 镍氢充电电池	1
6	7.4V/2200mAh 锂电池（带充电保护板）	1

7.1 用 3D 打印机打印出机器人的机械结构

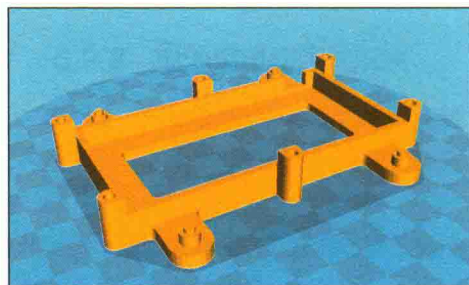
4 足机器人由身体（上、下盖）、关节、连接件、脚等几部分构成，为了保证打印件强度，打印填充率设置为 50%，support type 设置为 touching bulidplate，

platform adhesion type 设置为 brim。

身体由上、下两部分组成，用于存放 7.4V 电池和控制板。在 Cura 软件中设置打印参数时，请务必按照图 7.1、图 7.2 所示的方向设置，以确保打印效果。

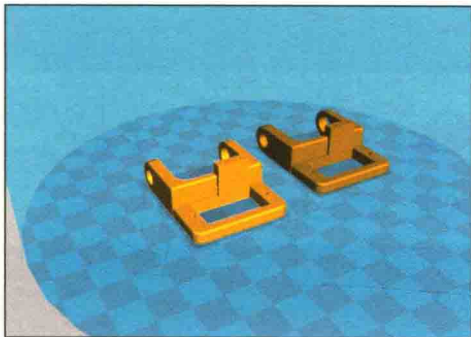


■ 图 7.1 身体上部分 3D 模型

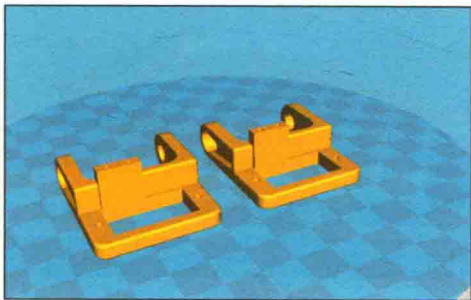


■ 图 7.2 身体下部分 3D 模型

关节分为左、右两组，每组两个相同部件，共4个打印件。打印时要各打印2份，如果想一次打印完成，可以按如图7.3、图7.4所示设置：在Cura中选中关节模型，单击鼠标右键，在弹出的快捷菜单中选择“Multiply object”，接着在“Number of copies”对话框中设置1，这样就会出现两个一模一样的模型了。



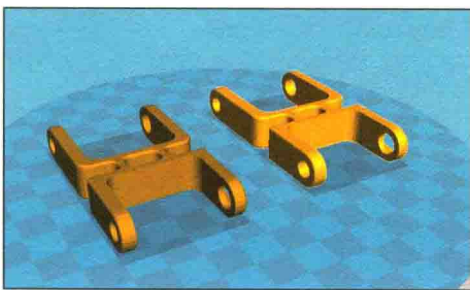
■ 图 7.3 左关节 3D 模型



■ 图 7.4 右关节 3D 模型

关节连接件用于连接关节和脚，按上述方法设置，同时打印2份，如图7.5所示。

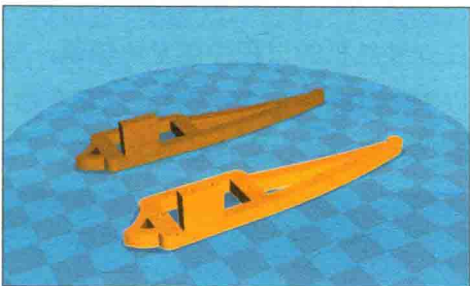
四脚也分为左、右两组，按图7.6、图7.7所示方式每组同时打印2份。



■ 图 7.5 关节连接件 3D 模型



■ 图 7.6 左脚 3D 模型



■ 图 7.7 右脚 3D 模型

7.2 舵机测试

在组装舵机与3D打印部件前，一定要确定所用的舵机在电气性能上是完好的，分别将舵机与Arduino的数字口2连接，刷入下列程序，你将会看到舵机在 0° 至 180° 之间来回摆动。

```
#include <Servo.h>
Servo myservo;
int pos = 0;
void setup() {
  myservo.attach(2);
}
void loop() {
  for (pos = 0; pos <= 180; pos
+= 1) {
    myservo.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos
-= 1) {
    myservo.write(pos);
    delay(15);
  }
}
```

7.3 零件组装

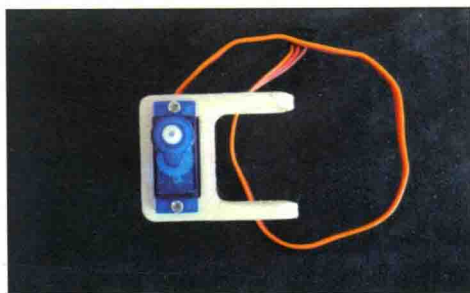
① 取出一个脚部零件和一个舵机，按下图所示方式组装起来，如果在安装舵机的过程中感觉比较紧，可以稍微打磨一下脚部零件。注意左脚、右脚与舵机的安装方向不要装反。



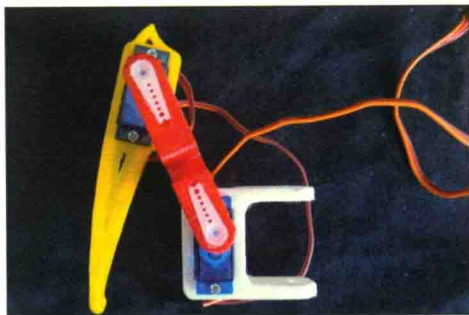
② 取出一个关节连接件，与上一步的组合件按下图所示方式进行组合，并用单向舵盘固定（不上螺丝）。



③ 取出关节件和舵机，将舵机放入关节件的舵机孔位，并用小螺丝固定。舵机位置与方向如下图所示

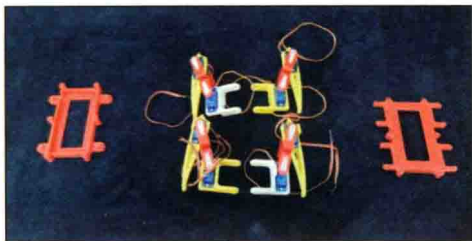


④ 将上两步的部件组合起来，并用舵盘固定，但不要上螺丝。这样，我们就完成了4足中的一足，其他3足的安装方法与此类似。



注：程序中使用了 flexitimer2 库，如果没有安装此库，编译时会报错。请下载 flexitimer2 文件夹，并将该文件夹复制到 Arduino IDE 的 libraries 文件夹下。

- 5 将4足与身体按如下方式组合，把7.4V 电池放入身体上、下夹板中，并用螺丝固定。



- 6 将舵机的数据线与 Arduino 控制板的数字口进行连接，请务必按照下图和表 7.2 所示顺序的连接舵机与 Arduino 控制板。

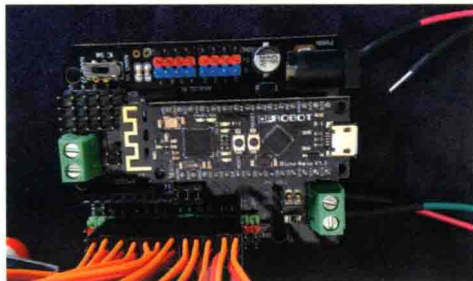


舵机对应数字口连接图

表 7.2 舵机与 Arduino 控制板数字口的对应关系

序号	名称	Arduino 控制板数字口
1	左上脚	D3
2	左上连接	D2
3	左上关节	D4
4	左下脚	D10
5	左下连接	D8
6	左下关节	D9
7	右上脚	D7
8	右上连接	D5
9	右上关节	D6
10	右下脚	D13
11	右下连接	D11
12	右下关节	D12

- 7 舵机的3根数据线的中，黄色线为数据，红色线为电源+，褐色线为GND。



7.4 安装电池

4 足机器人使用了 Arduino Nano 控制板和 I/O 扩展板，我们分别对其进行供电。Arduino Nano 采用 9V 镍氢充电电池（见图 7.8）供电，扩展板 PWM 供电使用 7.4V 电池（见图 7.9）供电，上步安装过程已经介绍。



图 7.8 9V 镍氢充电电池和转换插头

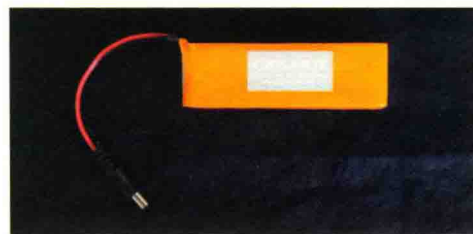
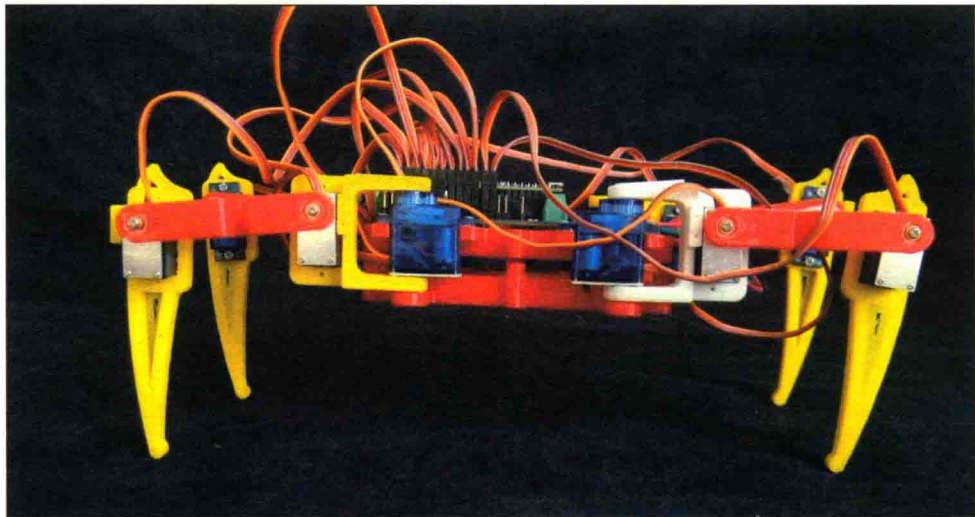


图 7.9 PWM 供电

7.5 初始化舵机位置

把“legs_init”代码上传到 Arduino 控制板，激活舵机回归中位，然后断开电源，

将各关节按图 7.10 所示方式摆放好，并拧紧所有舵盘上的螺丝。

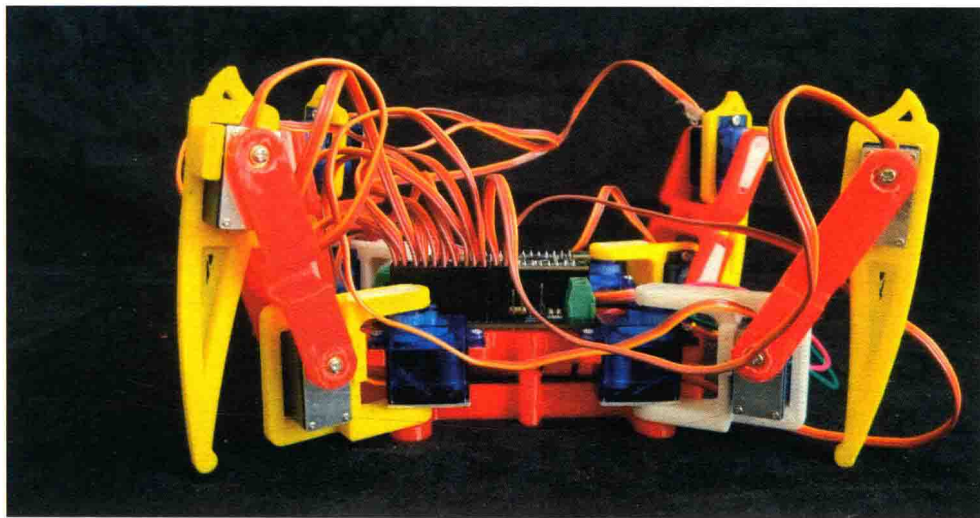


■ 图 7.10 初始化舵机后，将机器人摆成图中所示姿势再固定螺丝

```
#include <Servo.h>
Servo servo[4][3];
const int servo_pin[4][3] = { {2, 3, 4}, {5, 6, 7}, {8, 9, 10}, {11, 12, 13} };
void setup()
{
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            servo[i][j].attach(servo_pin[i][j]);
            delay(100);
        }
    }
    while (1);
}
void loop(void)
{
}
```

此时我们已经完成了所有的硬件安装，手动将蜘蛛的各关节摆放成如图 7.11 所示的姿态，也就是蹲下去的姿态，将“Go”代码上传到 Arduino 控制板，即可看到 4 足机器人执行下面的动作：

（1）站起来，等待 2s；（2）向前走 5 步，等待 2s；（3）后退 5 步，等待 2s；（4）右转，等待 2s；（5）左转，等待 2s；（6）挥舞爪子，等待 2s；（7）收回爪子，等待 2s；（8）坐下，等待 2s；（9）回到第 1 步的状态。



■ 图 7.11 运行“Go”程序的初始姿态

至此，4 足机器人脱机版就制作完成了，打开电源，让你的 4 足机器人走起来吧。

08

用 3D 打印部件制作跳舞机器人 BOB



想不想在你的书桌上放一个呆萌可爱的小小机器人呢？在无聊的时候，控制它来为你“跳个舞”，缓解工作压力或烦躁的心情。今天给大家分享的案例是精舞堂 BOB，它是一个根据前方物体的距离远近改变舞蹈动作的 3D 打印机器人。制作所需的器材见表 8.1。

表 8.1 项目器材

名称	数量
1.75mm PLA 3D 打印耗材	500g（橙色）
Beetle 控制器	1
Beetle 控制器扩展板	1
SG90 舵机	4
Risym HC-SR04 超声波模块	1
3.7V 锂电池	1
3.7V 锂电池充电器	1
单芯杜邦线	10 根（母-母）

本次打印件使用了 DFRobot 的新 3D 打印机 Overlord，其打印质量比上一代产品 DreamMaker 有了质的飞跃，打印件无论视觉上还是触觉上都比之前精细了很多（见图 8.1）。



图 8.1 3D 打印部件对比

8.1 3D 打印模型

本次制作的 3D 打印机器人由 4 部分组成：头部、底板、腿部、脚部。经过多次测试 BOB 的部件强度和在运动过程中的稳定性、可靠性，并平衡打印所需的时间、所消耗的原材料，对 3D 打印的部件设置了不同的填充率：头部为 20%，底板为 20%，腿部为 100%，脚部为 50%（见图 8.2~图 8.5）。这样一方面保证了机器人整体的稳定性，不至于出现头重脚轻的情况；另一方面腿部也有足够的强度，能承受住压力。

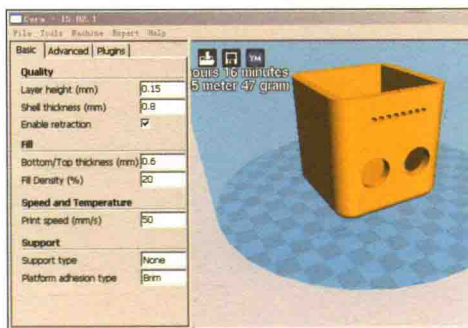
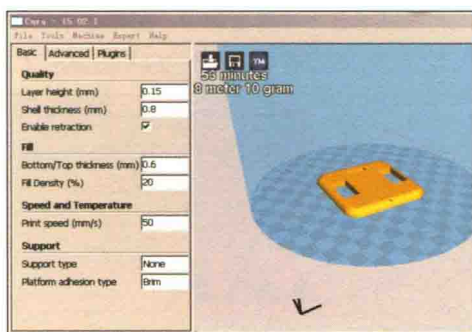
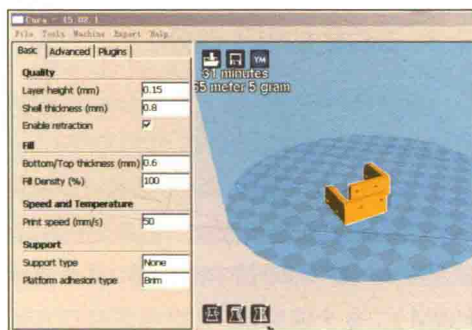


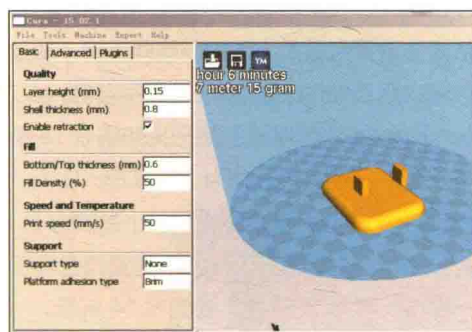
图 8.2 头部打印参数设置



■ 图 8.3 底板打印参数设置



■ 图 8.4 腿部打印参数设置



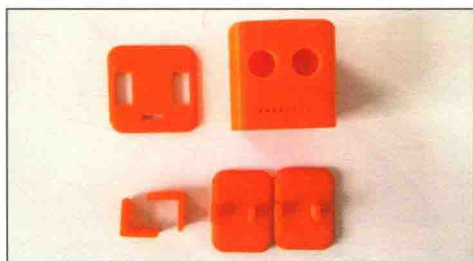
■ 图 8.5 脚部打印参数设置

8.2 机械结构组装

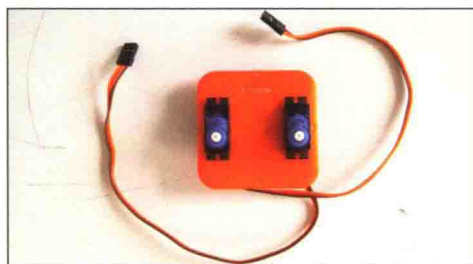
通过 3D 打印机打印出 4 个部分的组件：头部和底板各 1 个，腿部和脚部各 2 个（见图 8.6）。

取出两只舵机，按图 8.7 所示方向安装，

作为机器人腿根关节控制部分。打印件需要使用锉刀适当修模，简单清理一下舵机的槽位，以便轻松地安装舵机。

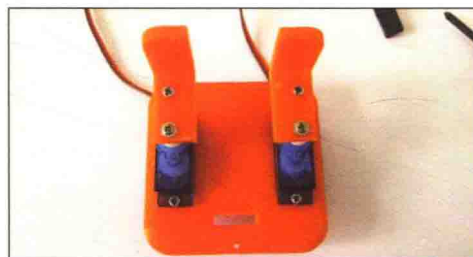


■ 图 8.6 打印出的部件



■ 图 8.7 将舵机安装到底板上

取出舵机的单向舵盘，按图 8.8 所示的方向安装好，然后用螺丝将其固定在腿部件上。固定舵盘旋转轴的地方要使用较长的螺丝，以保证舵机在转动过程中腿部能够与它同步转动。



■ 图 8.8 安装单向舵盘和腿部

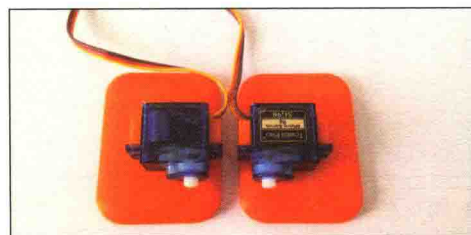
脚部机械部分将安装在上述组合件上。用螺丝将两个单向舵盘固定在腿部的垂直面内侧，并从内侧用小螺丝固定舵盘（见图

8.9)。请注意单向舵盘的方向，靠近脚的是旋转轴的位置。



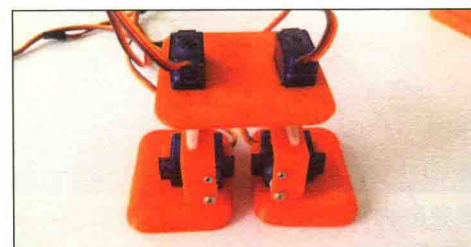
■ 图 8.9 用螺丝将两个单向舵盘固定在腿部的垂直面内侧

取出两个舵机安放在两只脚上，如图 8.10 所示。



■ 图 8.10 将两只舵机安放在两只脚上

将腿部组合件与左右脚部分组合到一起，并用长螺丝固定（见图 8.11）。此时 BOB 下半身机械结构的安装已经完成。请注意，以上旋转轴部分的长螺丝都不要拧得太紧，后面还要调整舵机的初始位置。



■ 图 8.11 组合腿部和脚部

头部的安装比较简单，只要按正确的方

向将头部卡在下半身上即可。整体效果如图 8.12 所示，可以看到 BOB 已经初具雏形了。

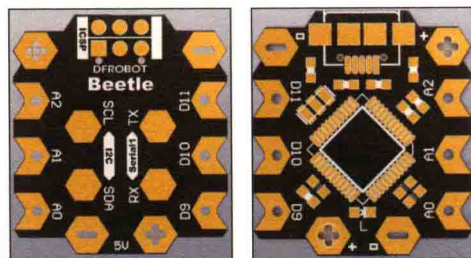


■ 图 8.12 按正确的方向将头部卡在下半身上

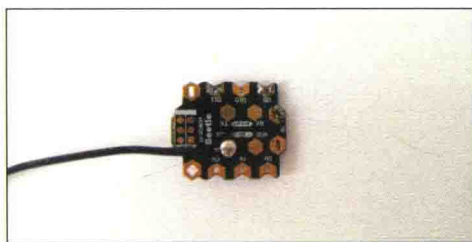
8.3 BOB 的电子部分

此次推荐使用的控制板是 Beetle 控制器和 Beetle 控制器扩展板。作为 DFRobot 全新一代 Arduino Leonardo 兼容微型控制器，Beetle 只有硬币大小，但是它的功能依然强大，有 10 个数字口（其中 4 个支持 PWM）、5 个模拟口、2 组电源端口，可以满足用户对 Arduino 的高要求。各位小伙伴在制作一个轻巧、美观的互动作品时，常常会为选择合适大小的控制器而苦恼，现在你可以把 Beetle 隐藏到任何作品当中。

Beetle 控制器的引脚定义如图 8.13 所示，其中 PWM 口分别为 D3、D9、D10、D11。注意：在 Beetle 扩展板上是没有 D3 的，D3 对应 SCL，因此我们需要自己焊接一根线，将 SCL 引出，如图 8.14 所示。

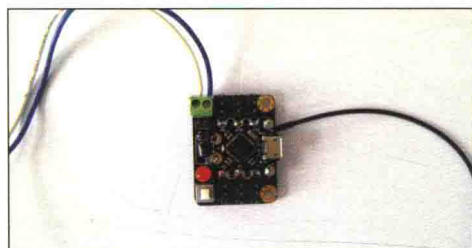


■ 图 8.13 Beetle 的引脚定义



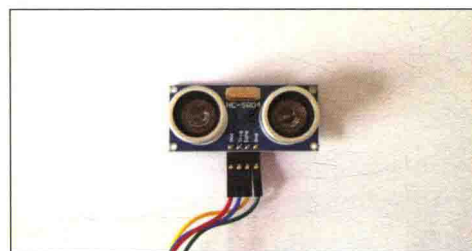
■ 图 8.14 将 SCL 引出

再将 Beetle 控制器安装在 Beetle 扩展板上, 依次焊接各引脚, 如图 8.15 所示。



■ 图 8.15 将 Beetle 主控器安装在 Beetle 扩展板上

BOB 的眼睛使用的是 HC-SR04 超声波测距传感器, 它的 4 根引脚 VCC、TRIG、ECHO、GND 分别用红、蓝、黄、绿色杜邦线引出 (见图 8.16)。



■ 图 8.16 HC-SR04 超声波测距传感器

舵机连线: 控制信号方面, 左脚舵机接 D11, 右脚舵机接 D3 (引出的 SCL 引脚), 左腿舵机接 D10, 右腿舵机接 D9; 电源方面, 所有 VCC 和 GND 接入扩展板。

超声波传感器连线: VCC、GND、Echo、Trig 引脚分别接扩展板的 VCC、GND、A0、A1。

连接好以后, 烧录一下程序, 调整舵机初始位置为 90° , 将下面的测试代码烧录进 Beetle 控制器中 (注意 Beetle 控制器的 Board 类型为 Arduino Leonardo)。

```
#include <Servo.h>
//creating the servo objects for
front, rear and mid servo
Servo myservorhip;
Servo myservolhip;
Servo myservorankle;
Servo myservolankle;
//setting the servo angle to 90°
for startup
byte LlegAngle = 90;
byte RlegAngle = 90;
byte LnetAngle = 90;
byte RnetAngle = 90;
//Setup function
void setup(){
  myservorhip.attach(9);
  myservolhip.attach(10);
  myservorankle.attach(3);
  myservolankle.attach(11);
  //move servos to center
  position -> 90°
  myservorhip.write(LlegAngle);
  myservolhip.write(RlegAngle);
  myservorankle.write(LnetAngle);
  myservolankle.write(RnetAngle);
  delay(2000);
}
void loop(){
}
```

烧录上述程序以后, 将调整 BOB 的腿部和脚部, 使其成为直立姿态。

然后烧录下列程序, 该程序的功能为: 当超声波测距测得的距离值小于 100 时, 脚部舵机转动, BOB 做 45° 角倾斜动作; 当距离值大于 100 时, BOB 左右腿扭动舞蹈。

```
#define ECHOPIN A0
//Pin to receive echo pulse
#define TRIGPIN A1
//Pin to send trigger pulse
#include <Servo.h>
// 声明调用 Servo.h 库
Servo myservorhip; // 创建一个舵机对象
Servo myservolhip; // 创建一个舵机对象
Servo myservorankle;
// 创建一个舵机对象
Servo myservolankle;
// 创建一个舵机对象
int pos = 0;
// 变量 pos 用来存储舵机位置
void setup(){
    Serial.begin(9600);
    pinMode(ECHOPIN, INPUT);
    pinMode(TRIGPIN, OUTPUT);
    myservorhip.attach(9);
    // 将引脚 9 上的舵机与声明的舵机对象连接起来
    myservolhip.attach(10);
    myservorankle.attach(3);
    myservolankle.attach(11);
}
void loop(){
    myservorankle.write(90);
    myservolankle.write(90);
    myservorhip.write(90);
    myservolhip.write(90);
    digitalWrite(TRIGPIN, LOW);
    //Set the trigger pin to low for 2μs delayMicroseconds(2);
    digitalWrite(TRIGPIN, HIGH);
    //Send a 10μs high to trigger ranging delayMicroseconds(10);
    digitalWrite(TRIGPIN, LOW);
    //Send pin low again
    int distance = pulseIn(ECHOPIN, HIGH);
    //Read in times pulse
    distance= distance/58;
    //Calculate distance from time of pulse
    if (distance<100){
        for(pos = 90; pos < 120; pos += 1){
            myservolhip.write(pos);
            myservorhip.write(pos);
            delay(30); // 延时 30ms 让舵机转到指定位置
        }
        // 舵机从 180° 转回 0°, 每次减小 1°
        for(pos = 120; pos>=90; pos-=1) {
            myservorhip.write(pos);
```

```

    myservolhip.write(pos); // 写角度到舵机
    delay(30);
    // 延时 30ms 让舵机转到指定位置
}
}
else{
    for(pos = 80; pos>=60; pos-=1) {
        myservorankle.write(pos);
        // 写角度到舵机
        myservolankle.write(pos);
        // 写角度到舵机
        delay(30);
        // 延时 30ms 让舵机转到指定位置
    }
    for(pos = 60; pos < 80; pos += 1){
        myservorankle.write(pos);
        // 写角度到舵机
        myservolankle.write(pos);
        // 写角度到舵机
        delay(30);
        // 延时 30ms 让舵机转到指定位置
    }
}
Serial.println(distance);
delay(50); //Wait 50ms before next ranging
}

```

8.4 测试运行

将 3.7V 锂电池的正负极插在 Beetle 扩展板的电源接口上，按下开关，BOB 会根据前方物体的距离，跳出不同的舞蹈。

有些初学者在使用 Beetle 时，加载新程序后，可能在电脑上再也找不到 Beetle 的虚拟串口。在排除接线问题后，此现象一般是新程序影响了 USB 通信导致。Beetle/Arduino Leonardo 的 USB 转串口是由相应的软件实现的（在 Arduino IDE 中实现），它和用户程序都在 ATmega32U4 中，如果用户程序影响了 USB 的相关操作，将导致看不到 USB 虚拟串口。Beetle 上电启动时将直接运行用户程序，因此通过

插拔 USB 不能使其进入 Bootloader。由于 Beetle 已经和电脑失去联系，不能自动复位，需要重新加载可用的用户程序（比如 Blink），以人工复位方式进行加载，具体方法是：在 Arduino IDE 中打开 blink 程序，单击“upload”上传程序，当 IDE 中显示“uploading”时，通过跳线短接 ICSP 接口中的 RESET 和 GND，然后断开，等待程序加载完毕，Beetle 复位完成，打开设备管理器，便可见对应的串口。

8.5 总结拓展

本次 BOB 的外形比较中规中矩，如果你有不错的创意，可以发挥出来，打造个性化的 BOB（见图 8.17）。另外在功能上你

也可以进行拓展，本例只是给 BOB 制定了两种动作，你可以修改程序，玩出更多的花样，比如让 BOB 伴随音乐起舞（加个 MP3 播放模块），或者换块 Bluno Beetle 控制板来用蓝牙遥控 BOB。



■ 图 8.17 发挥创意，打造个性化 BOB

09

让智能手机变身为会走路的机器人



澳大利亚有一位视频游戏开发者 Kevin Chan 设计了一款非常独特的 3D 打印机器人。这个机器人的闪亮之处在于，它的主体和控制中心是一部普通的智能手机！Kevin Chan 把自己创造的这个机器人命名为“MobBob”。智能手机本身就有内置的摄像头、话筒、指南针、触摸屏、陀螺仪、GPS、扬声器和 Wi-Fi 连接等功能，这样做既节省了电子器材的成本，同时也节约了开发功能的成本。而手机与机器人身体的连接，仅需要通过简单的蓝牙开启。最最关键的是这位老兄“开源了”，让我等“小白”也有机会体验一把造物的乐趣，还不快来？一起造起来！

制作所需器材见表 9.1，3D 打印设置见表 9.2，各 3D 打印模型如图 9.1~ 图 9.7 所示。

表 9.1 项目器材

序号	名称	数量
1	Bluno Nano 控制板（集成蓝牙 4.0）	1
2	Arduino Nano I/O 扩展板	1
3	TowerPro SG90 舵机	4
4	3.7V 电池	1
5	锂电池充电器	1
6	9V 电池	1
7	9V 电池转换插头	1
8	M3×15	4
9	卷式结束保护带（6mm）	1

表 9.2 模型文件打印设置

Quality	Layer height: 0.15
	Shell thickness: 0.8
Fill	Bottom/top thickness: 0.4
	Fill density: 50
Support	Support type: touching buildplate
	Platform adhesion type: brim

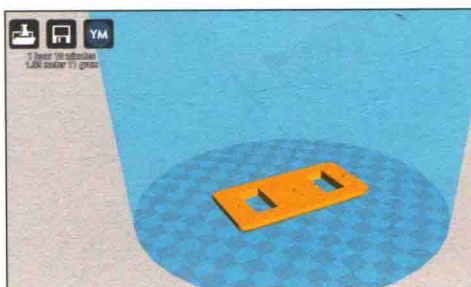


图 9.1 膝部卡槽

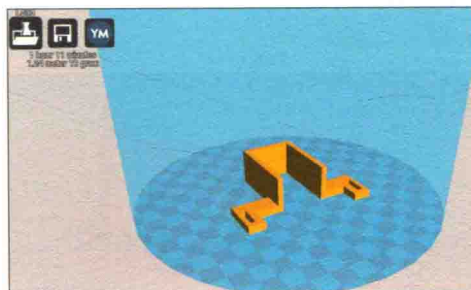
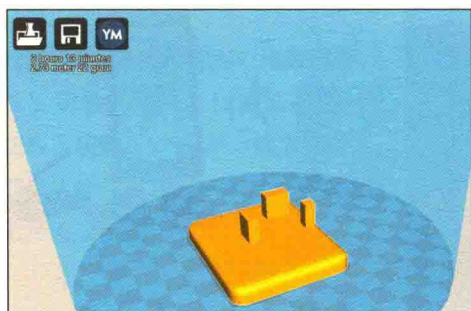
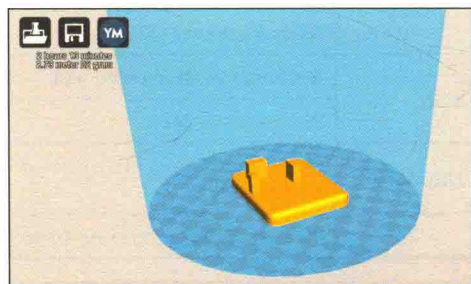


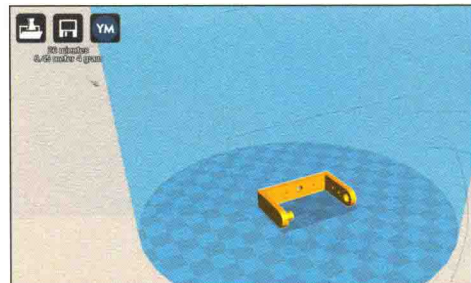
图 9.2 电池卡槽



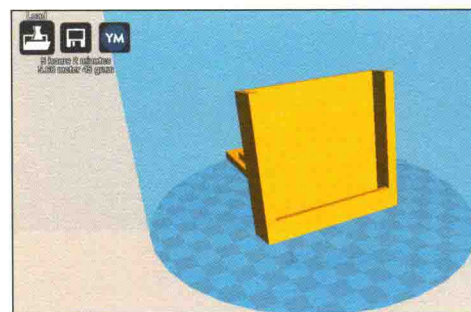
■ 图 9.3 左脚



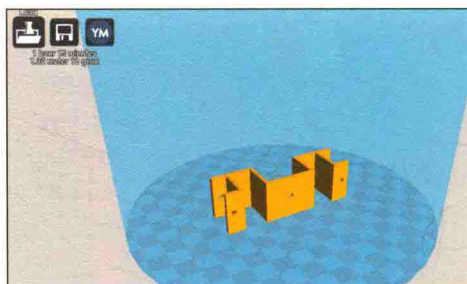
■ 图 9.4 右脚



■ 图 9.5 腿部



■ 图 9.6 手机卡槽（请实际测量手机宽度再进行打印）



■ 图 9.7 舵机卡槽（请务必按图中所示方位进行打印）

9.1 部件组装

准备好的所有部件如下图所示。

1



拿出舵机、1 个脚部部件和 1 个腿部部件。

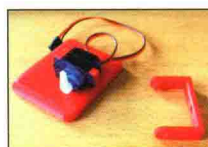
2



3

将舵机装在脚部部件上，请

尽量将舵盘按 90° 安装。



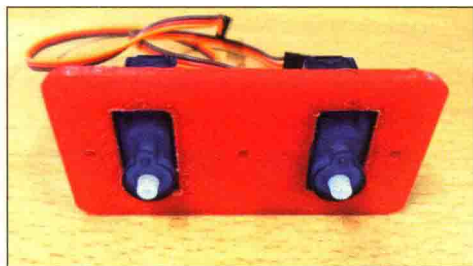
- 4 在腿部部件上方安装长舵盘，注意将来要用螺丝在里侧安装小舵盘的为机器人正面。



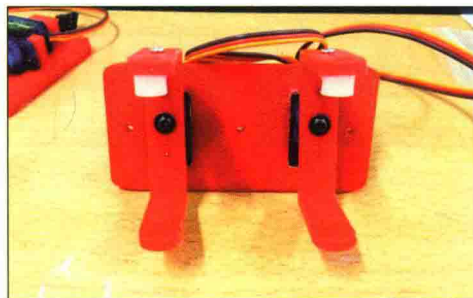
- 5 将脚部舵机的舵盘用螺丝固定到腿部部件侧面。



- 6 将舵机头朝下安装入膝部卡槽，注意有白色转动轴的一方是机器人正面。



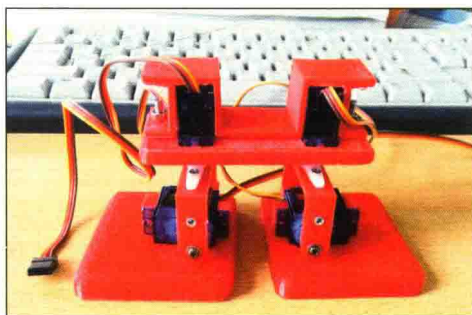
- 7 先将腿部与膝盖部分用螺丝固定好。



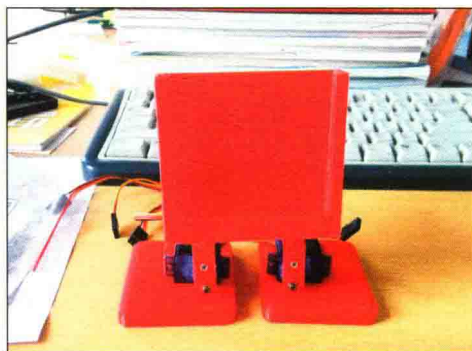
- 8 将上一步中的膝腿组合体分别与左脚、右脚组装起来，部件方向前面已经提示过。

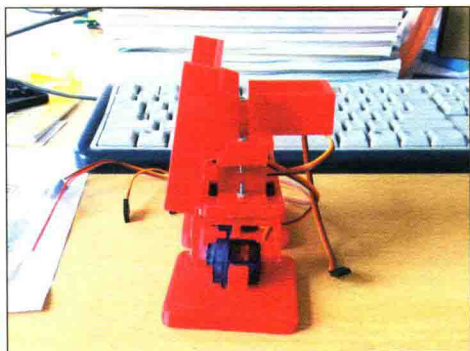


- 9 再将舵机卡槽卡在膝部舵机上方，用 M3 × 15 螺丝固定。



- 10 在以上组合体上依次增加手机卡槽和电池卡槽，用 M3 × 15 螺丝固定。





9.2 电路连接

先将 Bluno Nano 和 Nano I/O 扩展板叠加（注意 Bluno Nano 引脚的方向，Nano I/O 扩展板的引脚见图 9.8），4 个舵机与数字口的对应关系分别为：左膝舵机接 D6、左脚舵机接 D7、右膝舵机接 D8、右脚舵机接 D9。Servo power 分别接 3.7V 电池的正、负极，为 4 个舵机供电。9V 电源通过转接头与扩展板连接供电。电路部分的连接到此就完成了（见图 9.9），最后别忘记将你的手机安放到手机插槽中。

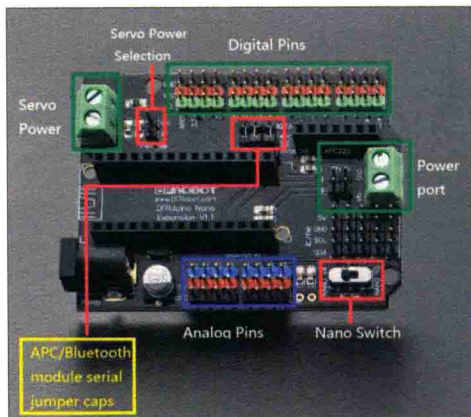


图 9.8 Nano I/O 扩展板的引脚

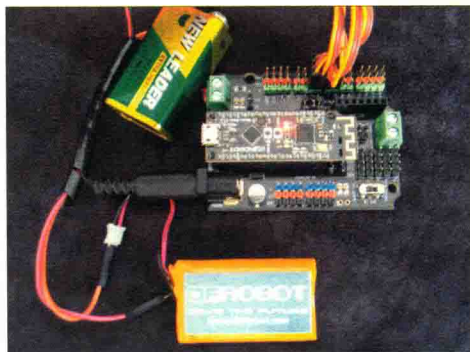


图 9.9 电路连接

9.3 程序代码和手机 App

在 Bluno nano 中刷入以下代码，再在手机上安装 MobBob 手机 App，激动人心的时刻就来到了。开启手机蓝牙，运行手机 App，选项如图 9.10 所示。

如果蓝牙配对成功，Bluno Nano 上的蓝牙指示灯会亮起。进入 App 界面后从向上滑动，会出现如图 9.11 所示的操作界面，你的智能手机就变身成为机器人了，后面的操作都很直观，快玩起来吧！

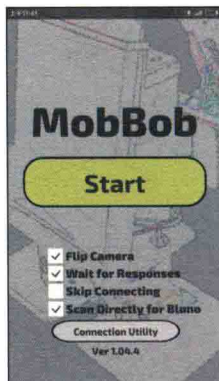


图 9.10 MobBob 手机 App

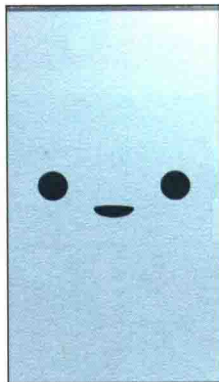


图 9.11 MobBob 机器人启动界面

第 3 章

制作个性化装备



- 10 智慧之光——3D 打印钥匙置物架
- 11 牛角灯帽
- 12 基于蓝牙通信的自行车“驴友”装备
- 13 点滴报警系统

10

智慧之光——3D 打印 钥匙置物架

回想刚开始玩 3D 打印时，曾经为了打印机调平、喷头堵等问题而“狂躁不安”；更曾为了成功打印出一个口哨而“欣喜若狂”。从 3D 打印完全的门外汉，到今天可以自己设计、打印一些小玩意，这一年多里感慨颇多、收获颇多。

玩 3D 打印时间长了，打印的东西也就慢慢多了起来。对待这些宝贝，总得找个地方给它们安家，于是产生了用 3D 打印机制作置物架的念头。既然要做，咱就要做得与众不同，在置物架的外形选择上几经考虑，最后选择了钥匙的外形（见图 10.1），其寓意是解决问题的方法和窍门。

制作所需材料见表 10.1。



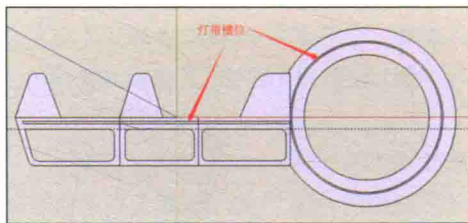
■ 图 10.1 智慧之光置物架

表 10.1 项目器材

1	Overlord Pro 3D 打印机
2	PLA 打印耗材（直径 1.75mm，黄色、粉色、蓝色、白色）
3	Arduino UNO 控制板
4	I/O 扩展板
5	炫彩 WS2812 LED 灯带，3m
6	人体红外热释电运动传感器
7	杜邦线若干

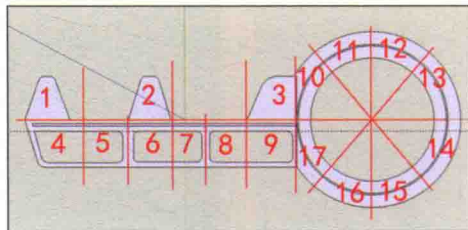
10.1 外形设计及 3D 打印

用 SketchUp 绘制图 10.2 所示的钥匙外形，注意是平面的哦，先别着急拉伸成立体的。因为受打印机可打印尺寸的限制，需要后续对其进行切分。



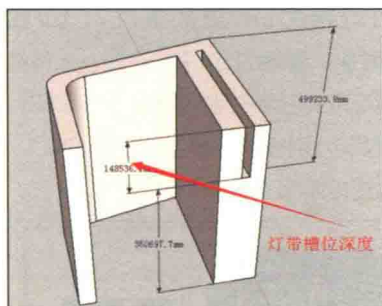
■ 图 10.2 置物架平面结构

对钥匙整体平面图形进行切分，如图 10.3 所示，按红色线将其切分为 17 部分。



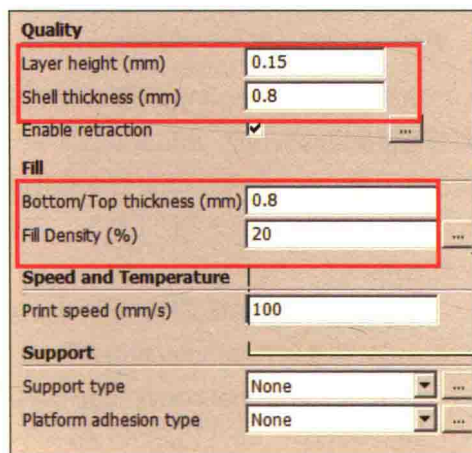
■ 图 10.3 切分方案

以切分后的第 4 部分为例，如图 10.4 所示，利用推、拉工具将其拉伸为下面的立体结构，注意灯带槽位的深度，每一个包含灯带槽位的部件都应保持一致。

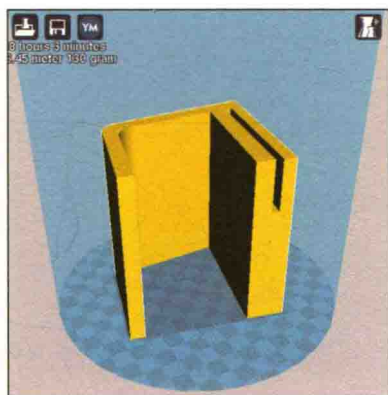


■ 图 10.4 第4部分拉升效果

考虑到打印部件的强度及自身重量问题，打印参数设置如图 10.5 所示，切片软件中的 3D 模型如图 10.6 所示。



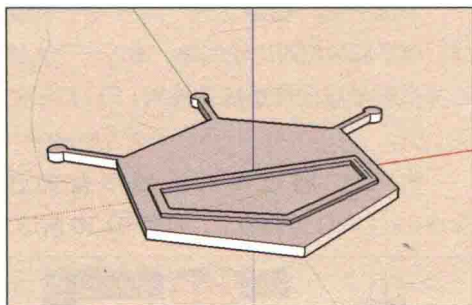
■ 图 10.5 打印参数设置



■ 图 10.6 3D 打印模型

其他部件的设置与此打印部件类似。当所有部件都打印完成后，使用 AB 胶将这些部件粘合起来，组成一把钥匙的形状。为了保证粘合的效果，可用锉刀对粘合的部分进行适当修正，保证结合处的美观。

由于笔者是 DF 创客论坛的版主，准备打印一些 DF 的标志作为装饰。用 SketchUp 建立模型，建模过程省略，都是基本图形，绘制好以后拉伸成立体的就 OK（见图 10.7）。



■ 图 10.7 DF 标志

再使用 3D 打印机打出来，当然，你可以用两种或者多样颜色进行打印，凭借 Overlord Pro 的断电续打功能，还可以打出双色标志，如图 10.8 所示。再用 AB 胶将它们粘贴在钥匙上，如图 10.9 所示。



■ 图 10.8 DF 标志打印实物

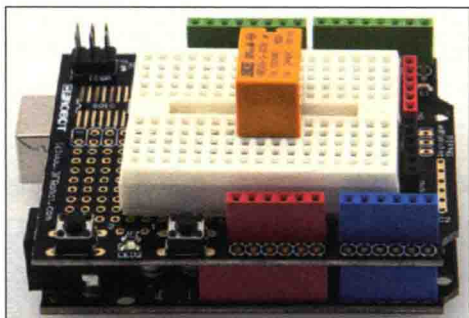


■ 图 10.9 贴标后的置物架

10.2 电子部分制作及安装

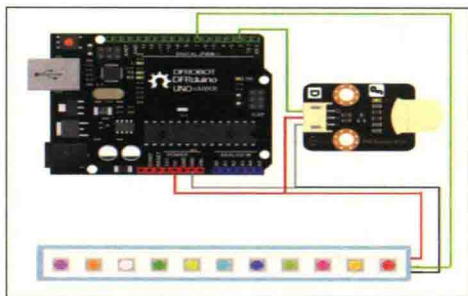
下面开始“智慧之光”电子部分的制作，电子部分相对比较简单，通过一个人体红外热释电运动传感器来控制LED灯带的亮、灭。

将I/O扩展板按对应线序叠加在Arduino UNO控制板上，如图10.10所示。



■ 图 10.10 主板叠加

LED灯带的Din接数字口7，5V和GND分别接扩展板的5V和GND。人体红外热释电运动传感器的数据线接数字口2，5V和GND分别接扩展板的5V和GND。整个电路连接如图10.11所示。



■ 图 10.11 电路连接

由于整捆的灯带长3m，而置物架中的灯带槽位是容纳不下的，所以要截取其中一段安放于槽位中，并清点LED灯珠的数目（当然，这是一项比较无聊的工作），不过后面有用到的。

下载下列地址中的库文件，将其更新到Arduino安装目录下libraries根目录下。

http://www.dfrobot.com/image/data/KIT0069/libraries_for_LEDStrip.zip

将下列代码烧录至Arduino控制器中，注意查看自己的灯珠数目，修改对应参数。

```
#include <Adafruit_NeoPixel.h>
#define PIN 7    // 灯带在Arduino上的对应接口
#define sensorPin 2    // 传感器在Arduino上的对应接口
#define LED_COUNT 81    // 灯带上灯珠的数目
Adafruit_NeoPixel leds = Adafruit_NeoPixel(LED_COUNT, PIN, NEO_GRB + NEO_KHZ800);

void setup()
{
    pinMode(7, OUTPUT);
    pinMode(sensorPin, INPUT);
    Serial.begin(9600);
}
```

```

    leds.begin();
    clearLEDs();
    leds.show();
}

void loop()
{
    digitalWrite(7, LOW);
    clearLEDs();
    byte state = digitalRead(sensorPin);
    if(state==1)
    {
        leds.begin();
        clearLEDs();
        leds.show();
        for (int i=0; i<LED_COUNT; i++)
        {
            rainbow(i);
            delay(50); //Delay between rainbow slides
        }
    }
    else
    {
        leds.begin(); //Call this to start up the LED strip
        clearLEDs(); //This function, defined below, turns all LEDs off...
        leds.show();
        digitalWrite(7, LOW);
    }
}

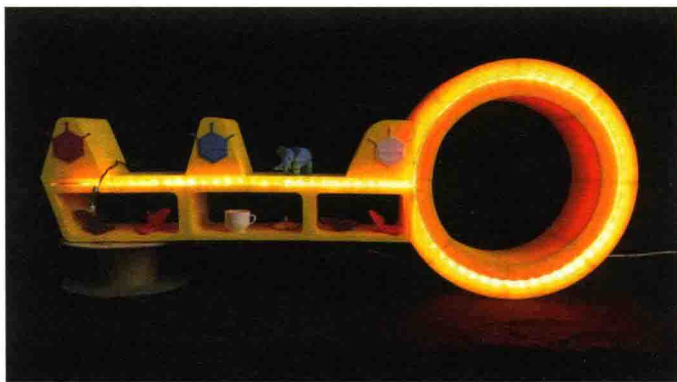
void clearLEDs()
{
    for (int i=0; i<LED_COUNT; i++)
    {
        leds.setPixelColor(i, 0);
    }
}

void rainbow(byte startPosition)
{
    int rainbowScale = 81 / LED_COUNT;
    for (int i=0; i<LED_COUNT; i++)
    {
        leds.setPixelColor(i, rainbowOrder((rainbowScale * (i + startPosition)) % 81));
    }
    leds.show();
}

```

```
uint32_t rainbowOrder(byte position)
{
  if (position < 31) //Red->Yellow(Red=FF,blue=0,green goes 00->FF)
  {
    return leds.Color(0xFF, position * 8, 0);
  }
  else if (position < 63) //Yellow->Green(Green=FF,blue=0,red goes FF->00)
  {
    position -= 31;
    return leds.Color(0xFF - position * 8, 0xFF, 0);
  }
  else if (position < 81) //Green->Aqua(Green=FF,red=0,blue goes 00->FF)
  {
    position -= 63;
    return leds.Color(0, 0xFF, position * 8);
  }
  else if (position < 81) //Aqua->Blue(Blue=FF,red=0,green goes FF->00)
  {
    position -= 81;
    return leds.Color(0, 0xFF - position * 8, 0xFF);
  }
  else if (position < 81) //Blue->Fuchsia(Blue=FF,green=0,red goes 00->FF)
  {
    position -= 81;
    return leds.Color(position * 8, 0, 0xFF);
  }
  else //160 <position< 191 Fuchsia->Red (Red =FF,green=0,blue goes FF->00)
  {
    position -= 81;
    return leds.Color(0xFF, 0x00, 0xFF - position * 8);
  }
}
```

测试效果如图 10.12 所示。如果你觉的这个玩法还不够好，完全可以使用 DF 的 Bluno 系列控制板，用手机来玩转灯带。



■ 图 10.12 效果测试

牛角灯帽

在西方的万圣节前夜，小孩会穿上化妆服，戴上面具，挨家挨户收集糖果，这可能是孩子们最开心的时刻了。作为创客爸爸，我当然要亲手为孩子们做一顶独特的万圣节牛角灯帽了。

此次制作的牛角灯帽，利用蓝牙 4.0 无线 BLE 编程实现蓝牙主控端和蓝牙从控端的数据通信，用触摸延迟开关遥控帽子上的牛角灯的开关，并加了声音提示功能。



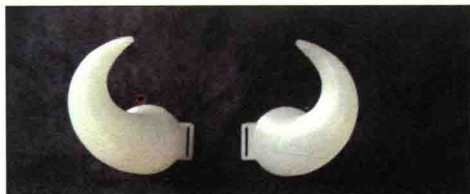
制作所需的器材见表 11.1。

表 11.1 项目器材

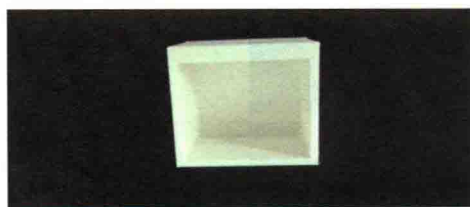
序号	名称	数量
1	Bluno Beetle 控制板 	2
2	LED 	4
3	数字蜂鸣器模块 (Arduino 兼容) 	1
4	数字触摸开关 Touch (Arduino 兼容) 	1
5	品胜 9V/250mAh 镍氢充电电池 	2
6	公母、母母杜邦线	若干
7	热熔胶枪、热熔胶棒	若干
8	PH2.54 排母端子套件，带金属弹片	1

11.1 3D 打印部分的制作

① 打印牛角部分。



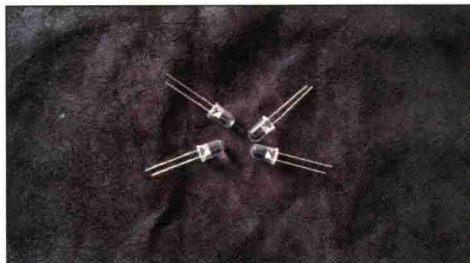
② 打印遥控器盒子。



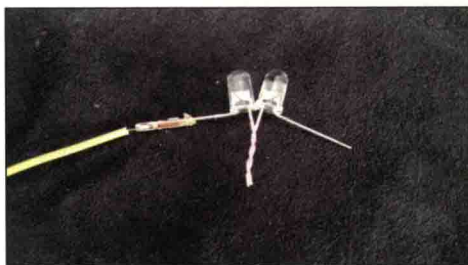
11.2 电路部分安装测试

11.2.1 LED 焊接安装

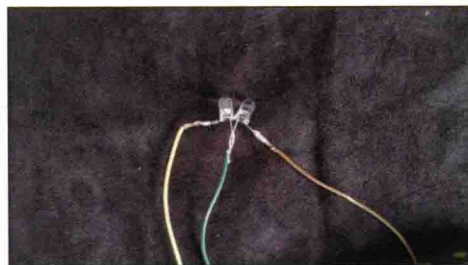
① 此次使用的 LED，其电气参数如下：工作电压 2.4~2.7V，最大电流 20mA，其中短针为负极，长针为正极，颜色为白色和蓝色。为了保证在夜晚的光亮程度，我们在每个牛角中放置 2 个 LED。



② 由于此次使用 Bluno Beetle 控制板，相对来说 GND 针脚少，因此 4 个 LED 负极共用一个 GND，焊接方式如下图所示。



③ 而对于 LED 的正极连线，考虑到焊接的牢固性，推荐大家使用 PH2.54 排母端子套件（带金属弹片）来连接 LED 正极和杜邦线，使其成为一个整体。



④ 这样一个牛角的 LED 有 2 根数据线、1 根负极线。为了保证 3 根线的牢固性，同时又保证相互之间不要搭线，我们在焊接完成后，使用热熔胶枪打上了一层热熔胶。



5 为了保证 LED 在牛角中不会晃来晃去，可以将一些泡沫材料填充在牛角中。



6 这样一只牛角已经制作完毕，另外一只的制作方法与之类似。

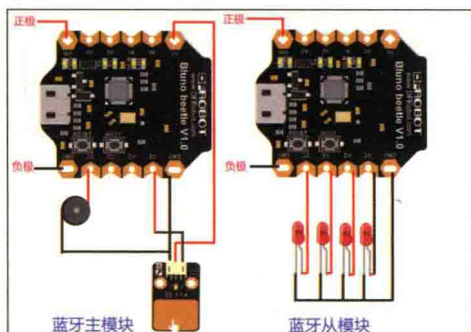


11.2.2 Bluno Beetle 焊接

1 根据此项目的需求，我们首先需要对主、从模块的供电线进行焊接，注意，外部供电正极输入为 VIN，负极为 GND，红色线焊接在 VIN 上，黑色线焊接在 GND 上。



2 对于蓝牙主控端，我们需要焊接 D2、D5、GND、+5V 引脚；对于蓝牙从模块，焊接 D2、D3、D4、D5、GND、+5V 引脚，具体连线如下图所示。



11.2.3 蓝牙配对

通过 AT 指令配置 BLE 设备，固件 1.8 版本之后 AT 开关失效，进入 AT 指令模式需要先把串口监视器的右下角的调为“没有换行符”，然后在串口监视器中输入“+++”，然后在串口监视器中输入“+++”，进入 CMD 模式，即 AT 模式。具体操作方式如下。

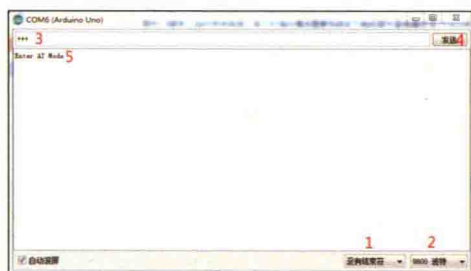


图 11.1 进入 AT 模式

(1) 打开 Arduino IDE。

(2) 选择菜单“tools”→“Serial Monitor”。开启串口监视器。

(3) 在下面的两个下拉菜单中分别选择“No line ending”(见图 11.1 的 1) 和“115200 baud”(见图 11.1 的 2)。

(4) 在最上面的输入框中输入“+++”(见图 11.1 的 3)，并单击“Send”按钮(见图 11.1 的 4)。

(5) 如果收到“Enter AT Mode”(见

图 11.1 的 5)，就证明已经进入 AT 指令模式了。

按下列 AT 指令来设置蓝牙主从模块。

设置 BLE 工作在主机状态: AT+ROLE=ROLE_CENTRAL<CR+LF>。

设置 BLE 工作在从机状态: AT+ROLE=ROLE_PERIPHERAL<CR+LF>。

如果设置成功，在串口监视器中会出现“OK”，此时分别对主从模块上电，配对成功后，可以看到 LINK 灯亮。

11.2.4 代码烧录

主模块代码

```
int key=5;
void setup()
{
    Serial.begin(115200);
    pinMode(key,INPUT);
    // 将 2、5 号数字口设置为输入状态，将 13 号数字口设置为输出状态
    pinMode(2,OUTPUT);
}
void loop()
{
    int n =digitalRead(5);
    // 创建一个变量 n，将 5 号数字口的状态采集出来赋值给它
    if (n==1)
    // 判断 n 是否为高电平，如果是则执行下面的语句，不是则跳过
    {
        digitalWrite(3,HIGH);
        tone(2,200);
        delay(500);
        Serial.println(n);
    }
    else
    {
        digitalWrite(3,LOW);
        noTone(2);
    }
}
```

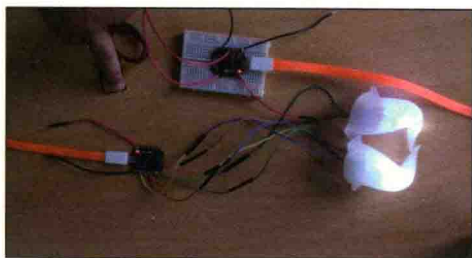
从模块代码

```
void setup()
{
    pinMode(2,OUTPUT);
    pinMode(3,OUTPUT);
    pinMode(4,OUTPUT);
    pinMode(5,OUTPUT);
    Serial.begin(115200);
}

void loop()
{
    int n=0;
    char a;
    a=Serial.read();
    n=a-48;
    if (n==1)
    // 判断 n 是否为高电平，如果是执行下面的语句，不是则跳过
    {
        digitalWrite(2,HIGH);
        digitalWrite(3,HIGH);
        digitalWrite(4,HIGH);
        digitalWrite(5,HIGH);
        delay(500);
        Serial.println(n);
    }
    else
    {
        digitalWrite(2,LOW);
        digitalWrite(3,LOW);
        digitalWrite(4,LOW);
        digitalWrite(5,LOW);
    }
}
```

11.3 总装

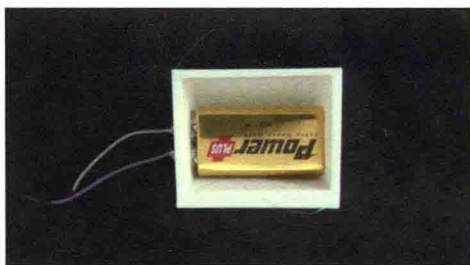
- 1 在总装前，需要分别对蓝牙主从模块及设备进行测试。



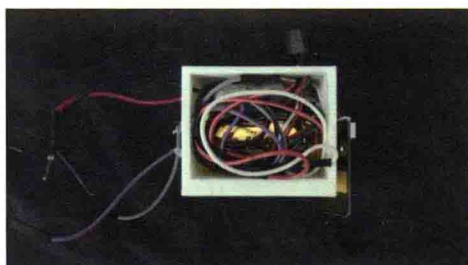
- 2 考虑到牛角的牢固性，我使用针线把牛角缝在帽子上，具体缝纫方法不在此处详细介绍。



3 遥控器先安装电池。



4 将控制板、蜂鸣器、触摸开关安装在遥控器盒子中。



好了，现在你可以戴着你的牛角帽参加活动了。上述教程中使用蓝牙 BLE 编程实现了数据之间的简单通信，大家也可以将灯光的效果做得复杂些，玩出更多花样。

12

基于蓝牙通信的自行车“驴友”装备

作为一名喜欢骑自行车的“驴友”，我除了考虑给自己的爱车添加各种炫酷的装备外，最大的需求莫过于研究如何让骑行过程变得更加安全，例如骑行时如何让身后的车辆及行人提前得知自己的转向，夜间行驶如何照明，如何得知当前的环境温度/湿度，在野外环境下如何准确得知自己的地理位置等。基于蓝牙通信的自行车“驴友”装备便

应运而生了。

本作品以旅行包为载体，利用蓝牙技术实现车头主控端与背包转向指示器之间的通信，具有转向指示、车头照明、环境温度/湿度检测、GPS 定位等功能。同时车头主控端是可拆卸的，当“驴友”下车后，可将车头控制端缚在手臂上继续使用。

制作所需的器材见表 12.1。

表 12.1 项目器材

序号	名称	数量
1	Arduino Mega2560 (Arduino UNO)	3
2	DF-Bluetooth V3 蓝牙模块	2
3	I/O 传感器扩展板 V7.1	2
4	I2C LCD1602 液晶模块 (兼容 Arduino Gadgeteer)	1
5	输入扩展板 V2.0 (集成摇杆、按键)	1
6	DFRduino GPS Shield-LEA-6H	1
7	炫彩 WS2812 LED 灯带	1
8	10W 高亮 LED 灯珠 (暖白色、120° 角)	1
9	数字继电器模块 (Arduino 兼容)	1
10	DHT11 温度/湿度传感器	1
11	LCD Keypad Shield 按键扩展板	1
12	电池组	3
13	USB 转串口模块	1

蓝牙主、从模块程序的流程如图 12.1、图 12.2 所示。

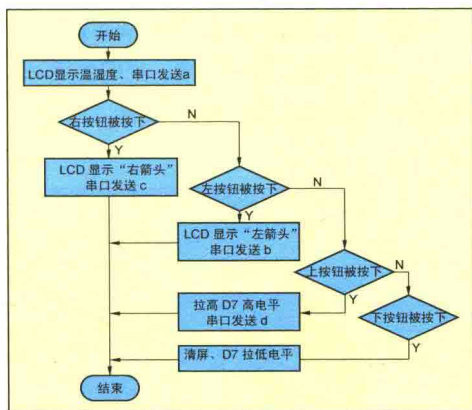


图 12.1 蓝牙主模块（车头主控端）程序流程图

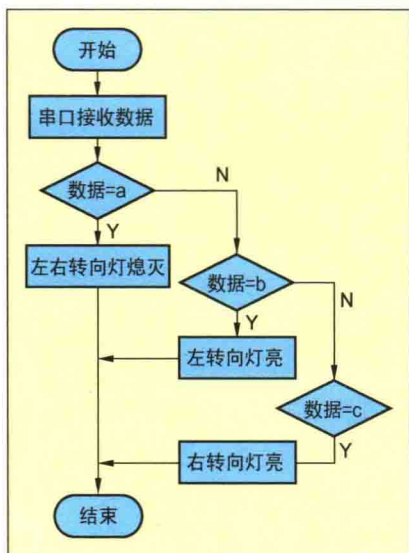


图 12.2 蓝牙从模块（从控端）程序流程图

12.1 机械结构打印

在这个作品中，我们使用到了 3 部分机械结构，分别是转向器灯槽、车头照明灯灯罩、车头手柄控制端卡槽。我们使用 3D 打印机打印出其机械结构，如图 12.3~图 12.5 所示。

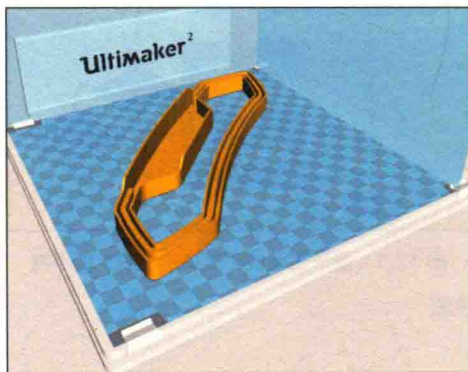


图 12.3 转向器灯槽（填充率 30%+Brim）



图 12.4 车头照明灯灯罩（填充率 30%+Brim）

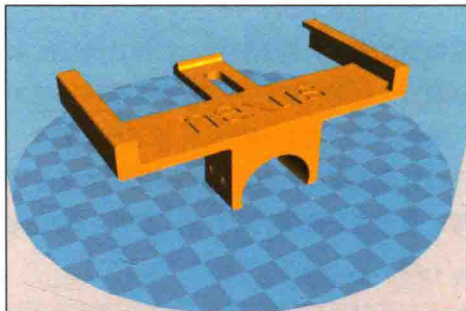


图 12.5 手柄控制端卡槽（填充率 50%+Brim）

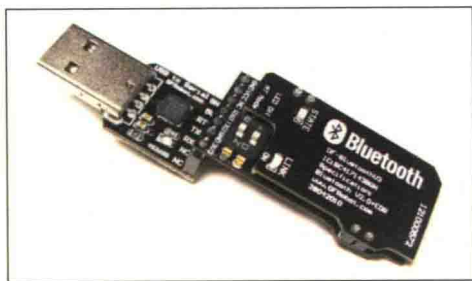
12.2 电路结构安装

12.2.1 蓝牙通信模块设置

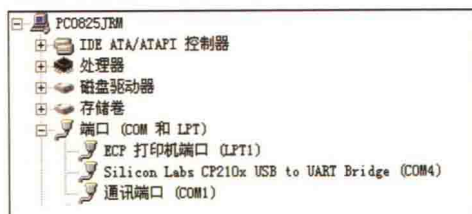
1 切换蓝牙模块到 AT 模式：如下图所示，模块有一个 2 位拨码开关，“LED Off”是 LINK 灯的开关（拨到标有“ON”的一端为开，拨到无字一端为关），可以关闭 LINK 灯以省电；“AT Mode”是 AT 命令模式开关，拨到标有“ON”的一端进入 AT 命令模式，拨到无字一端退出 AT 命令模式。



- ② 将 DF-Bluetooth V3 蓝牙串口模块按接口顺序插在 USB 转串口模块上。



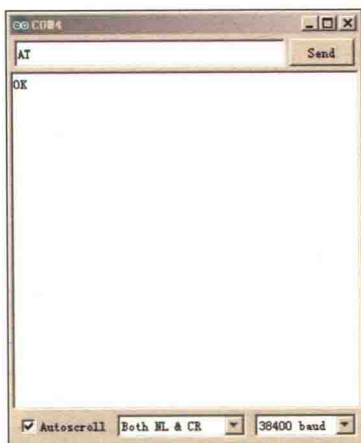
- ③ USB 转串口模块需要安装驱动程序，安装成功后，会在设备管理器中显示为一个串口，这里是下图中的 Silicon Labs CP210x USB to UART Bridge (COM4)。这个 COM 口是用来配置蓝牙模块的。



- ④ 我们需要用串口监视器通过串口指令来配置蓝牙模块。有很多串口调试软件可以使用，当然也可以用 Arduino IDE 自带的串口监视器。打

开 Arduino IDE，查看是否显示串口，我们这里选择 COM4，打开串口监视器界面，设置波特率为 38400，选择“换行和回车模式”（Both NL&CR）。设置完成后，在串口中输入“AT”（不区分大小写），测试蓝牙模块与 USB 串口是否建立连接，如果连接成功，会显示“OK”。按同样的方法在串口输入相关的 AT 指令就能进行配置了。

注意：在 AT 模式中波特率默认为 38400，和通信时设置的波特率无关。当 AT 指令设置完毕后，将“AT Mode”开关拨到另一端退出 AT 命令模式，重新上电后设置才生效。



- ⑤ 两块蓝牙模块进行配对，需要通过“AT+ROLE=1”指令将其中一个设置为主模块，通过“AT+ROLE=0”指令将另一个设置为从模块。

12.2.2 转向器部分的电路安装

- 1 从炫彩 WS2812 LED 灯带中，截取两段长度适当的灯带，将其安放在打印出来的转向器灯槽中，并将接头用 3 根杜邦线引出。注意，在焊接时由于焊点相距较近，请勿让其相互搭线。



- 2 蓝牙从控端模块由 3 部分构成，底层为 Arduino Mega2560（可用 Arduino UNO 代替）。中间层为 I/O 传感器扩展板 V7.1，在 I/O 传感器拓展板上，叠加安装蓝牙从模块。将左、右 LED 的信号线分别插在从控端模块的数字口 8、12 上，然后将从控端模块放置于背包的夹层内。



- 3 我们将转向指示灯置于背包外部，通过两套尼龙螺丝固定在背包表面。

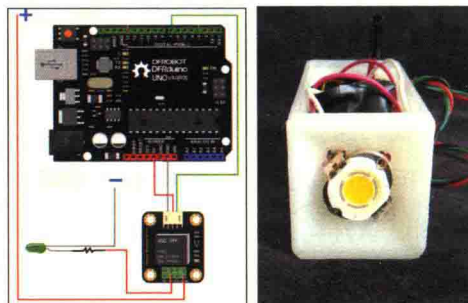


12.2.3 车头照明灯电路安装

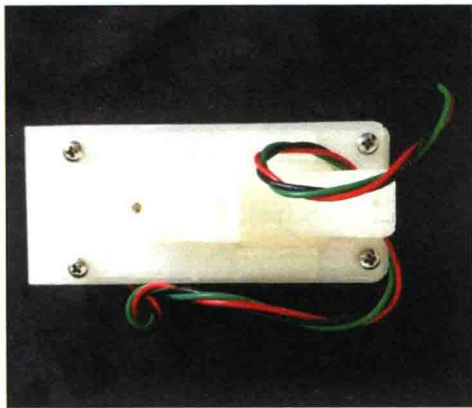
- 1 将 10W 高亮 LED 安装在车头灯罩打印件上，注意 LED 引脚带凹口的一端为负极。这款高亮 LED 灯珠内部包含 9 个 LED 阵列，可以用于普通的照明用途。



- 2 将继电器、电池与 LED 连成一个回路（图中的 +、- 分别代表电源的正极、负极），塞入车头灯罩打印件中。



- ③ 将3芯杜邦插头一端从灯罩侧面的孔中引出,然后盖上灯罩底板,上紧螺丝。

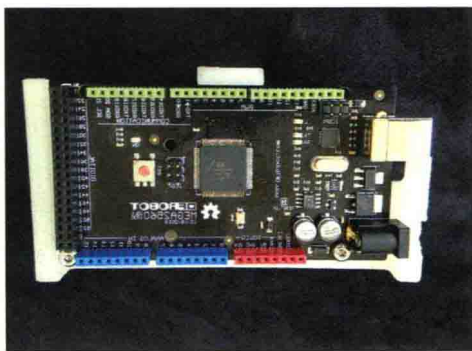


- ④ 最后用固定件将车灯固定在车把上。

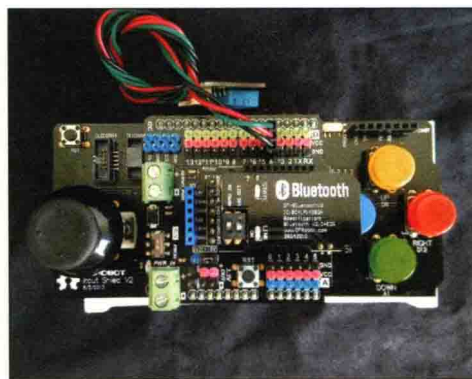


12.2.4 车头主控端电路安装

- ① 将 Arduino Mega2560 控制板安置于车头手柄控制端卡槽内,并用螺丝固定。由于在设计时没有考虑预留串口和电源供电口的位置,所以请自行修模,留出上述两个接口的位置。



- ② 在控制板上增加输入扩展板 V2.0, 这样我们就为主控端添加了控制按钮。在输入扩展板上增加 I/O 传感器扩展板 V7.1, 为照明灯和 DHT11、LCD1602 显示屏以及蓝牙主模块提供了模块化插接平台,避免了烦琐的插线。



- ③ 完成上述模块的连接后,我们可将其固定在自行车头。对于不同的车型,模型设计可能稍有不同,请针对具体的车型适当调整,以保证主控端安装的稳定性。



12.2.5 DHT11 和 LCD1602 显示屏的连接

1 DHT11 可实时侦测当前环境下的温度 / 湿度值，并将值显示在 LCD1602 显示屏上。DHT11 与 I/O 传感器扩展板的数字口 4 连接，车前灯与 I/O 传感器扩展板的数字口 7 连接。



2 LCD1602 显示屏在本作品中起到了显示信息的作用，既可以显示 DHT11 侦测到的温度 / 湿度值，又可以显示按钮按下后的状态变化。LCD1602 与 I/O 传感器扩展板的 I²C 口连接，请注意接线顺序，不要接反。



12.2.6 GPS 模块安装

这个作品中的 GPS 模块是相对独立的单元，因此它的安装相对简单。我使用 Arduino UNO、DFRduino GPS Shield-LEA-6H、LCD Keypad Shield 按键扩展板逐层叠加，实现 GPS 的定位。这里需要注意的是，GPS 增益天线需要外置，否则无法进行定位。



到目前为止，我们已经完成了所有的机械结构安装和电路连接，只需烧录程序就可以实现功能了。注意：烧录蓝牙主、从模块的程序时，一定要先将蓝牙通信模块移除，否则无法烧录；在烧录 GPS 模块代码时，也要先将所有的扩展板移除，等程序烧录至 Arduino UNO 中，再将扩展板叠加。

接下来，对主控端和从控端加电，测试一下你的“驴友”装备吧！

13

点滴报警系统

生病打点滴是再正常不过的事情了，但是有时因为需要输液的量比较大、时间比较长，可能会出现药液打完还没有及时拔出针头或更换新药液的情况。如果药液打完，针头还留在体内，会导致血液回流（见图 13.1），严重的甚至会有生命危险。点滴报警系统的创意，正是基于避免出现这种情况的目的。



■ 图 13.1 打点滴时血液回流

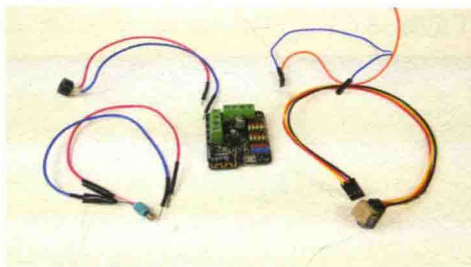
该作品由患者端和医护站两部分构成，如图 13.2 所示。患者端（见图 13.3）以一个 3D 打印盒子作为外壳，内置振动器、带功放的扬声器模块及 LED。当点滴袋内的药液面到达临界值时，安装在药袋上的非接触式液位传感器会发出控制信号，此时振动器振动、扬声器响起、LED 点亮，提醒患者药液已用完，需要拔出针头或更换药液。同时，佩戴在患者手上的报警器会向医护端发送报警信号。医护端（见图 13.4）接收到患者端信号后，功放扬声器模块响起，

LCD 屏幕显示“XX 号患者点滴已经输完”。

制作所需的器材见表 13.1。



■ 图 13.2 点滴报警系统



■ 图 13.3 患者端器材



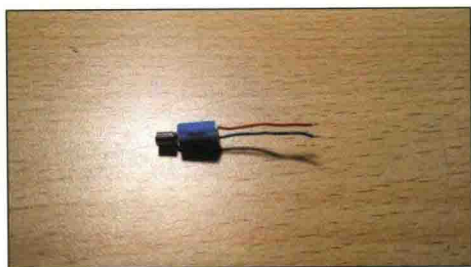
■ 图 13.4 医护端器材

表 13.1 项目器材

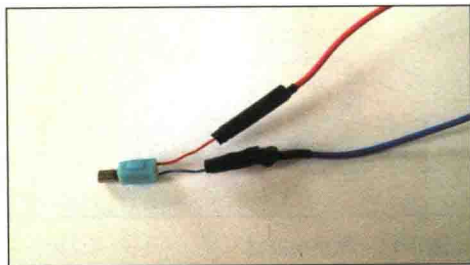
1	Bluno 控制板 × 1
2	带功放的扬声器模块 × 2
3	LCD Keypad Shield 按键扩展板 × 1
4	Romeo BLE mini 控制板 × 1
5	非接触式液位传感器 × 1
6	LED × 1
7	振动器（三星手机振动器）× 1
8	卷式结束保护带 × 1
9	高品质公母头跳线 21cm × 30 根
10	3D 打印耗材

13.1 振动器

由于三星手机振动器买来时连线较短（见图 13.5），我们需要对其进行延长。手机振动器的连线非常细，在焊接时一定要格外要小心，否则很容易断。焊接好后，用黑胶带将焊点包好，以免焊点搭线短路（见图 13.6）。



■ 图 13.5 三星手机振动器



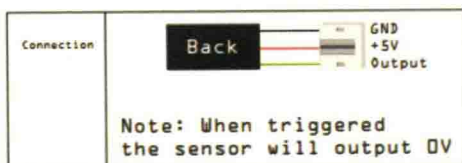
■ 图 13.6 延长振动器连线

13.2 非接触式液位传感器

在设计该作品时，曾经考虑使用导线来完成药液高度的检测，但是存在这样一个问题：导线在不同的药液里，可能会与药液发生反应，污染并破坏药液成分。而 DFrobot 的这款非接触式液位传感器（见图 13.7）就没有这个问题。它会在液面低于传感器所在水平面时发出信号。传感器不用接触溶液，这大大延长了传感器的使用寿命，同时避免污染药液。注意：液体容器壁必须是非金属的。这款传感器非常灵敏、稳定，输出电流可达 100mA，可直接驱动继电器，其引脚分布如图 13.8 所示。



■ 图 13.7 非接触式液位传感器

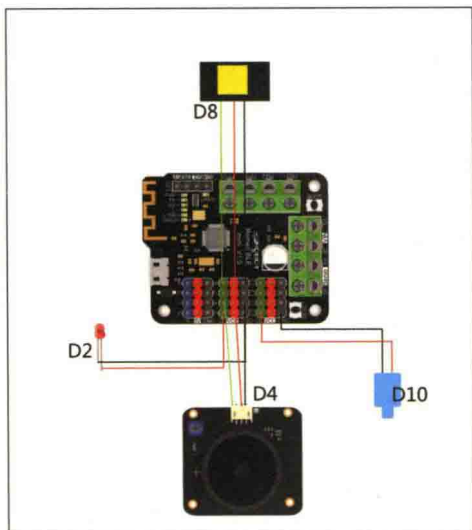


■ 图 13.8 非接触液位传感器引脚分布

13.3 硬件连接

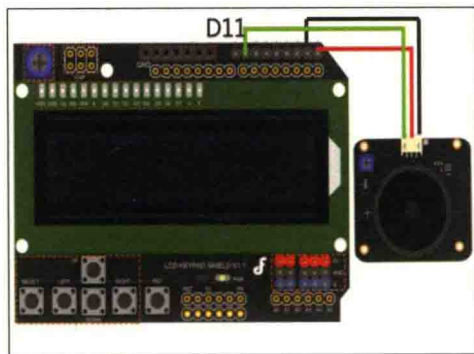
患者端：非接触式液位传感器接入 Romeo BLE mini 控制板的数字口 8，将振动器接入 Romeo BLE mini 的数字口 10，LED 接 Romeo BLE mini 的数字口 2，带

功放的扬声器模块接 Romeo BLE mini 的数字口 4，如图 13.9 所示。



■ 图 13.9 患者端电路连线

医护端：连接相对简单，将带功放的扬声器模块接 LCD Keypad Shield 按键扩展板的数字口 11，扩展板再插到 Bluno 控制板上即可，见图 13.10。

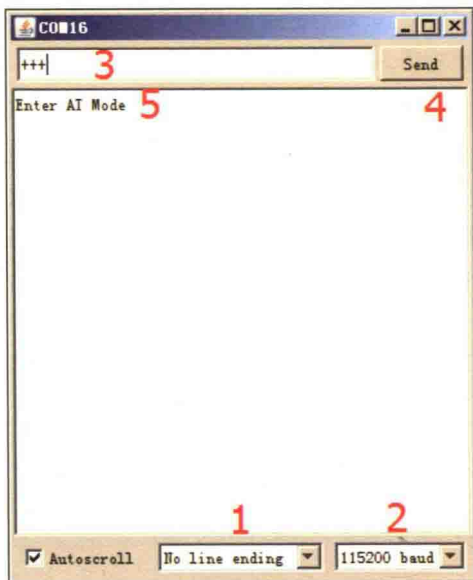


■ 图 13.10 医护端电路连接

13.4 蓝牙配对

通过 AT 指令配置 BLE 设备，进入 AT 指令模式需要先把串口监视器右下角调为

“No line ending”，然后在串口监视器中输入“+++”，进入 CMD 模式，即 AT 模式。具体操作方法如下。



■ 图 13.11 串口监视器

(1) 打开 Arduino IDE。

(2) 选择菜单“tools”→“Serial Monitor”，打开串口监视器。

(3) 在下面的两个下拉菜单中分别选择“No line ending”（见图 13.11 的 1）和“115200 baud”（见图 13.11 的 2）。

(4) 在最上面的输入框中输入“+++”（见图 13.11 的 3），并单击“Send”按钮（见图 13.11 的 4）。

(5) 如果收到“Enter AT Mode”（见图 13.11 的 5），就证明已经进入 AT 指令模式了。

按下列 AT 指令来设置蓝牙主从模块。

设置 BLE 工作在主机状态: AT+ROLE=ROLE_CENTRAL<CR+LF>。

设置BLE工作在从机状态: AT+ROLE=ROLE_PERIPHERAL<CR+LF>。

如果设置成功,在串口监视器中会出现“OK”,此时分别对主从模块上电,配对成功后,可以看到LINK灯亮。

13.5 代码编写

打开Arduino IDE,分别将下列两段代码烧录至患者端控制器和医护端控制器中运行,测试功能是否正常。

患者端程序

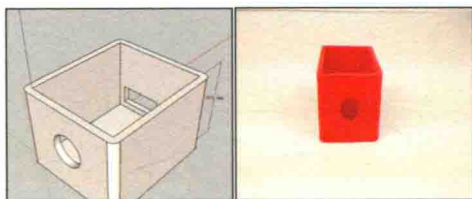
```
void setup()
{
    Serial.begin(115200);
    pinMode(8,INPUT);
    pinMode(10,OUTPUT);
    pinMode(4,OUTPUT);
    pinMode(2,OUTPUT);
}
void loop()
{
    int n =digitalRead(8);
    if (n==1)
    {
        Serial.println(n);
        digitalWrite(10,HIGH);
        digitalWrite(2,HIGH);
        tone(4,600);
        delay(1000);
    }
    else
    {
        digitalWrite(10,LOW);
        digitalWrite(2,LOW);
        noTone(4);
    }
}
```

医护端程序

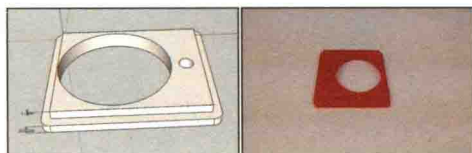
```
#include <LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
void setup()
{
    Serial.begin(115200);
    lcd.begin(16, 2);
    lcd.setCursor(0,0);
    lcd.print("Nursing station");
    pinMode(11,OUTPUT);
}
void loop()
{
    int n=0;
    char a;
    a=Serial.read();
    n=a-48;
    if (n==1)
    {
        lcd.clear();
        lcd.print("Hanging water");
        lcd.setCursor(0,1);
        lcd.print("The NO 1");
        tone(11,600);
        delay(1000);
    }
    else
    {
        lcd.begin(16, 2);
        lcd.setCursor(0,0);
        lcd.print("Nursing station");
        noTone(11);
        delay(1000);
    }
}
```

13.6 模型设计及打印

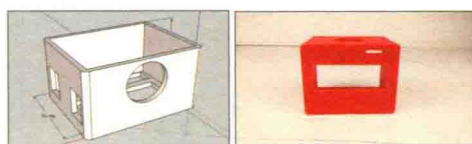
使用SketchUp建立患者端和医护端的3D模型文件,具体建模过程不再详细介绍。3D模型与打印后的实物如图13.12~图13.14所示。



■ 图 13.12 患者端盒子

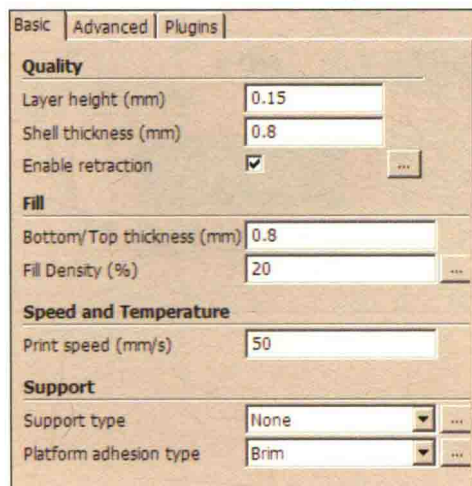


■ 图 13.13 患者端盒盖



■ 图 13.14 医护端

此次打印的模型是电子设备的外壳，没有过高的强度要求，对打印参数如图 13.15 所示设置即可。



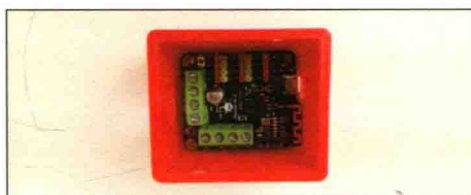
■ 图 13.15 打印参数设置

这里注意“Platform adhesion type”中选择了“Brim”，是为了保证在打印接触

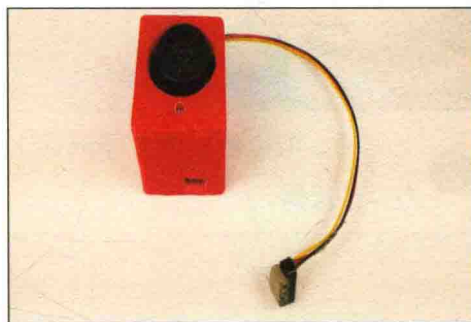
面比较大时，保证打印模型与平台的接触面不产生翘边。

13.7 作品安装

患者端：将 Romeo BLE mini 控制板安装到患者端盒子中（见图 13.16），并用螺丝固定。将 LED、振动器、带功放的扬声器模块、液位传感器与 Romeo BLE mini 控制板连接，将前三者放入盒子，并用盖子封装，液位传感器用连线拖在外面（见图 13.17）。



■ 图 13.16 安装患者端控制板



■ 图 13.17 安装传感器、LED 等设备

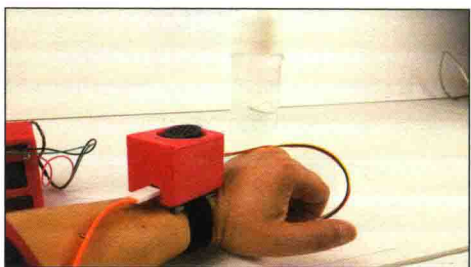
医护端：先将 LCD Keypad Shield 按键扩展板叠加在 Bluno 控制板上，再将其放置到医护端盒子内，调整位置，使 LCD 屏安放平整。最后将带功放的扬声器模块安装于盒子顶部，并与 LCD 扩展板的数字口 11 连接（见图 13.18）。



■ 图 13.18 安装医护端设备

13.8 作品测试

- ① 将患者端缚于患者手腕处。



- ② 上电后，液位传感器位于液面下方，不报警。



- ③ 医护端上电后的状态。



- ④ 当液位传感器位于液面上方时，患者端报警。



- ⑤ 医护端出现对应号码的报警信息及提示音。



第 4 章

制作物联网装置

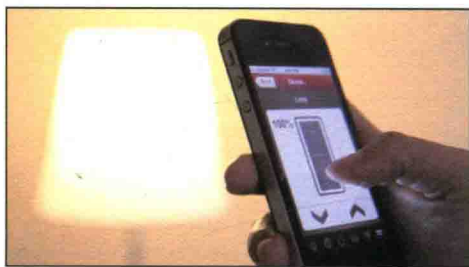


- 14 通过网络控制的灯
- 15 烟雾宝盒——基于 Yeelink 平台的二手烟雾监测报警装置
- 16 DIY 空气净化器

14

通过网络控制的灯

作为网络时代的居民，怎么能不了解物联网呢？物联网简单点说，就是物物相连的互联网。它是以互联网为核心延伸和扩展的网络，使得用户端延伸和扩展到了任何物品，物品之间能进行信息交换和通信。只要你的路由器连接了互联网，你就可以通过手机 App 远程控制你的灯，就像在家里控制一样（见图 14.1），不论你身处国内还是国外。



■ 图 14.1 通过 App 控制灯

今天我们就使用 Arduino+W5100 网络扩展板，结合 Yeelink 平台，实现通过手机 App 和网页远程控制 LED 灯的开关。

制作所需的器材见表 14.1。

表 14.1 项目器材

1	Overlord Pro 3D 打印机
2	PLA 3D 打印耗材（直径 1.75mm，白色）
3	Arduino UNO 控制板
4	W5100 网络扩展板
5	红色 LED
6	220Ω 电阻

14.1 用户注册

由于该作品是基于 Yeelink 平台的，所以首先需要登录 <http://www.yeelink.net> 进行用户注册。

14.2 添加设备和传感器

登录 Yeelink 平台，单击用户头像进入用户中心，然后单击“我的设备”→“添加新设备”进入设置设备界面，按要求填写有关设备的情况（见图 14.2）。



■ 图 14.2 添加新设备

完成新设备“WebLed”的添加后，单击“管理设备”进入“WebLed”设备管理界面（见图 14.3），下方有“添加传感器”按钮，可为该设备添加一个传感器并设置其属性。在本例中，传感器类型选择为“开关”，设置完成后保存即可。



传感器名
为它起一个好记的名字, 以区别于其它传感器

WebLed开关

类型
开关

标签TAGS
标签之间请使用逗号(,)分隔

WebLed开关

描述
不超过30个字符

Arduino+W5100

取消 保存

■ 图 14.3 添加传感器

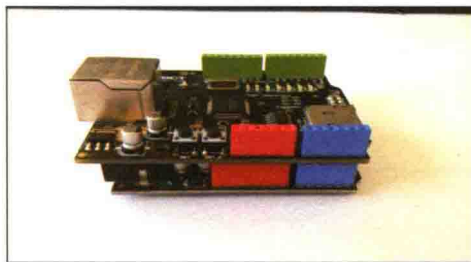
此时, 再次单击“管理设备”界面, 我们就新建了一个名为“WebLed”的设备, 并且为该设备添加了开关型传感器“WebLed 开关”。如果需要查看设备编号和传感器编号, 在图 14.4 所示的“状态 URL”中可见, 其中“device”后面跟的数字“344482”为设备编号, “sensor”后面跟的数字“382935”为传感器编号。这两个数字请记好, 后面我们还会用到。至此, 我们的前期平台注册工作已经结束, 下面开始连接线路。



■ 图 14.4 “状态 URL”中包含设备编号和传感器编号

14.3 电路连接

电路连接方法较为简单, 将 W5100 网络扩展板对应叠加在 Arduino UNO 上 (见图 14.5), LED 正极插在 Arduino 的数字口 5, LED 负极连接 220Ω 电阻, 然后插在 Arduino 的 GND 口。



■ 图 14.5 电路连接

14.4 加载库文件

进入 Yeelink 官网下载 Yeelink SDK, 下载完成后解压, 并将文件夹复制到 Arduino 安装目录下的 Libraries 目录。复制完成后, 重启 Arduino IDE, 即可使用 Yeelink 的库。

注意: 要将文件夹名中的下划线去掉, 否则会出现无法调用库的编译错误提示。解压到 Arduino 安装目录下的 Libraries 目录下的必须是根目录文件夹, 不能再包含文件

夹, 否则一样会出现无法调用库的编译错误提示。

14.5 烧录程序

将以下代码复制到 Arduino 编译器中, 注意修改用户 API KEY、设备号和传感器号, 然后将程序刷入控制板。API 编号可在“账户”→“我的账户设置”中查看(见图 14.6)。

```
#include <SPI.h>
#include <Ethernet.h>
#include <Wire.h>
#include <math.h>
byte buff[2]; //for yeelink api
#define APIKEY "a7ddf40c770aafd53d4d5df3161f215c" // 此处替换为你自己的 API Key
#define DEVICEID 344482 // 此处替换为你的设备编号
#define SENSORID1 382935 // 此处替换为你的传感器编号
//assign a MAC address for the ethernet controller
byte mac[] = { 0x00, 0x1D, 0x72, 0x82, 0x35, 0x9D};
//initialize the library instance:
EthernetClient client;
char server[] = "api.yeelink.net";
//name address for yeelink API
unsigned long lastConnectionTime = 0;
//last time you connected to the server, in milliseconds
boolean lastConnected = false;
//state of the connection last time through the main loop
const unsigned long postingInterval = 3*1000;
//delay between 2 datapoints, 30s
String returnValue = "";
boolean ResponseBegin = false;

void setup() {
  pinMode(5, OUTPUT);
  Wire.begin();
  //start serial port:
  Serial.begin(57600);
  //start the Ethernet connection with DHCP:
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    for(;;)
  }
  else {
```

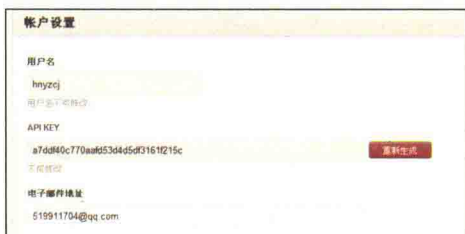


图 14.6 在“账户”→“我的账户设置”中查看 API KEY

```

    Serial.println("Ethernet configuration OK");
}
}

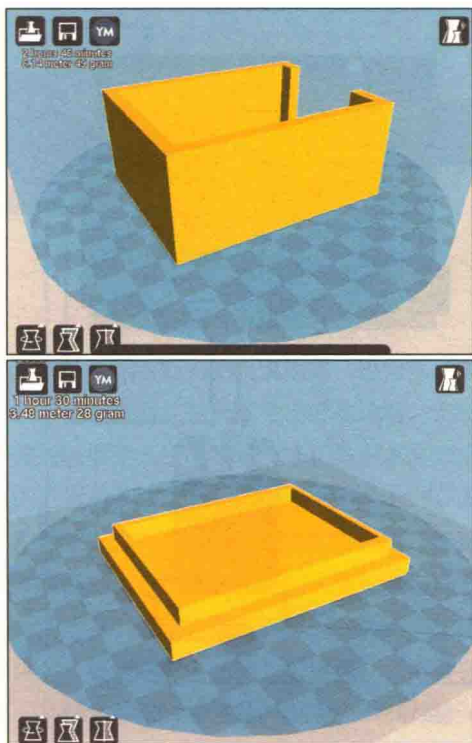
void loop() {
    //if there's incoming data from the net connection
    //send it out the serial port. This is for debugging
    //purposes only:
    if (client.available()) {
        char c = client.read();
        //Serial.print(c);
        if (c == '{')
            ResponseBegin = true;
        else if (c == '}')
            ResponseBegin = false;
        if (ResponseBegin)
            returnValue += c;
    }
    if (returnValue.length() != 0 && (ResponseBegin == false))
    {
        Serial.println(returnValue);
        if (returnValue.charAt(returnValue.length() - 1) == '1') {
            Serial.println("turn on the LED");
            digitalWrite(5, HIGH);
        }
        else if (returnValue.charAt(returnValue.length() - 1) == '0') {
            Serial.println("turn off the LED");
            digitalWrite(5, LOW);
        }
        returnValue = "";
    }
    //if there's no net connection, but there was one last time
    //through the loop, then stop the client:
    if (!client.connected() && lastConnected) {
        Serial.println();
        Serial.println("disconnecting.");
        client.stop();
    }
    //if you're not connected, and ten seconds have passed since
    //your last connection, then connect again and send data:
    if (!client.connected() && (millis() - lastConnectionTime >
postingInterval)) {
        //read sensor data, replace with your code
        //int sensorReading = readLightSensor();
        Serial.print("yeelink:");
        //get data from server
        getData();
    }
}

```

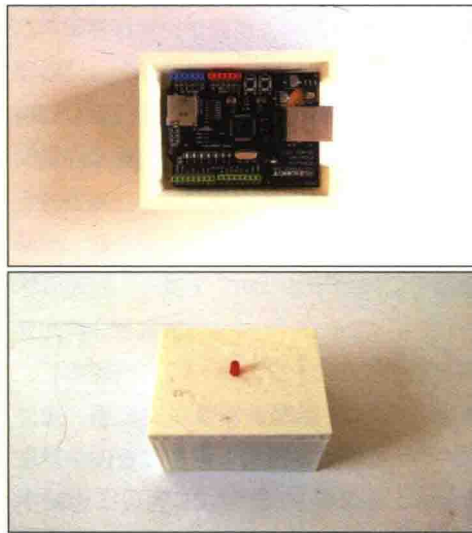
```
//store the state of the connection for next time through
//the loop:
lastConnected = client.connected();
}
//this method makes a HTTP connection to the server and get data back
void getData(void) {
  //if there's a successful connection:
  if (client.connect(server, 80)) {
    Serial.println("connecting...");
    //send the HTTP GET request:
    client.print("GET /v1.0/device/");
    client.print(DEVICEID);
    client.print("/sensor/");
    client.print(SENSORID1);
    client.print("/datapoints");
    client.println(" HTTP/1.1");
    client.println("Host: api.yeelink.net");
    client.print("Accept: *");
    client.print("/");
    client.println("*");
    client.print("U-ApiKey: ");
    client.println(APIKEY);
    client.println("Content-Length: 0");
    client.println("Connection: close");
    client.println();
    Serial.println("print get done.");
  }
  else {
    //if you couldn't make a connection:
    Serial.println("connection failed");
    Serial.println();
    Serial.println("disconnecting.");
    client.stop();
  }
  //note the time that the connection was made or attempted:
  lastConnectionTime = millis();
}
```

14.6 制作外壳

使用 SketchUp 为 WebLED 设计一个外壳, 具体模型文件如图 14.7 所示, 实际打印出的外壳如图 14.8 所示, 将叠加后的板子放入打印件中, 将红色 LED 固定在顶盖上。



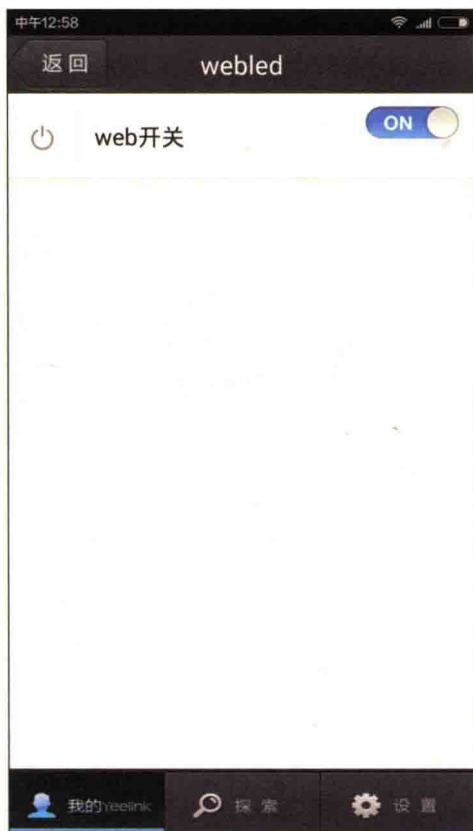
■ 图 14.7 WebLED 外壳 3D 模型



■ 图 14.8 实际打印出的外壳

14.7 测试运行

将网线插入 W5100 的网口，给 Arduino UNO 上电，分别用手机客户端（见图 14.9）、网页客户端（见图 14.10）打开和关闭 LED。



■ 图 14.9 手机客户端



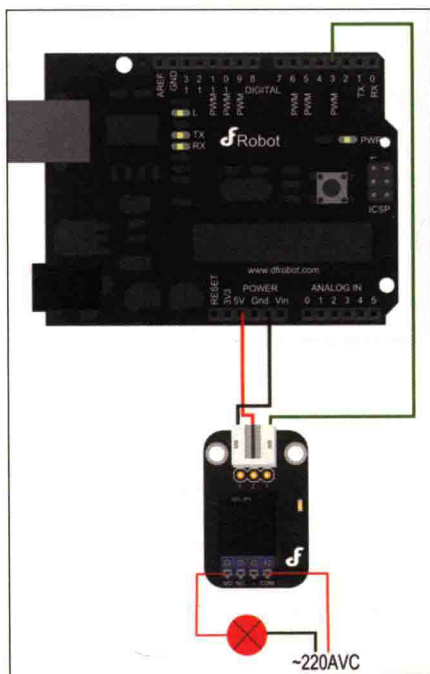
■ 图 14.10 网页客户端

14.8 任务拓展

也许你觉得这个 WebLED 不是很亮，完全可以通过图 14.11 所示的设备来做一盏更亮的 WebLED。图 14.11 中的灯与前面的 LED 没有本质区别，都是通过网络进行开关，只不过通过继电器外接了大电流设备。如果你胆子再大点，完全可以通过继电器控制交流供电的灯来玩玩，接线方法如图 14.12 所示，这里不再详细介绍。



■ 图 14.11 一盏更亮的 WebLED



■ 图 14.12 通过继电器控制交流供电的灯

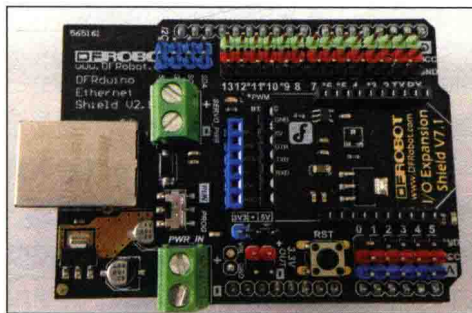
15

烟雾宝盒——基于 Yeelink 平台的二手烟雾监测报警装置

二手烟，也称为环境烟草烟（ETS），既包括吸烟者吐出的主流烟雾，也包括从纸烟、雪茄或烟斗中直接冒出的侧流烟，也就是指吸烟的人所呼出的气体和香烟本身燃烧时的烟雾的俗称。二手烟中包含包括 40 多种致癌物质和钋 210 等放射性物质，如被不吸烟的人吸进体内，会对人体健康造成很大的伤害。话说笔者常年受“老烟枪”的“熏陶”，经常给“老烟枪”提意见，“不要在公共场合吸烟”，但是，但是……“老烟枪”经常是这边答应了，那边好像又忘记了，继续在办公室吸烟。无奈笔者萌发了制作一个二手烟雾检测报警装置的想法，时刻提醒“老烟枪”。“老烟枪”如果在房间内点燃烟，可以让我们通过网络或者手机客户端了解房间里的烟雾值。更重要的是，当烟雾达到设定的报警值时，它可以通过发送微博方式提醒您，那个地方就不要去了。这就是本文向您介绍的——基于 Yeelink 平台的二手烟烟雾监测报警装置。

本制作以 DFRduino UNO R3 作为控制板，通过模拟烟雾传感器（MQ2），检测空气中的香烟浓度，并将数据通过 W5100 控制板（见图 15.1）实时传输到 Yeelink 平台，从而实现本地连接和网络报警。本地方式对烟雾值进行等级划分，烟雾值 <

500 时，亮起蓝灯，表示安全；烟雾值在 500 ~ 900 时，亮起黄灯，表示附近有烟雾；烟雾值 > 900 时，亮起红灯，表示烟雾浓度很大，并发出警报声。网络方式通过 Yeelink 网络平台和手机客户端实时监测烟雾浓度数据值，当烟雾值超过报警值时，通过发送微博方式报警。



■ 图 15.1 W5100 网络拓展板

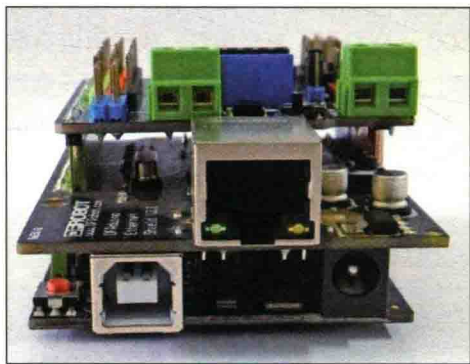
制作所需的器材见表 15.1。

表 15.1 项目器材

序号	名称	数量
1	DFRduino UNO R3 控制板	1
2	I/O 传感器扩展板 V7.1	1
3	W5100	1
4	模拟 CO 气体传感器 (MQ2)	1
5	数字 LED 发光模块 (Arduino 兼容)	1
6	数字蜂鸣器模块 (Arduino 兼容)	1
7	盒子	1

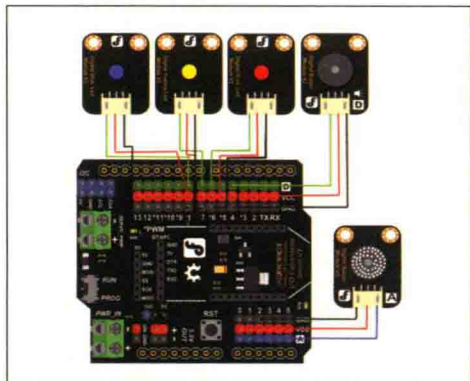
15.1 器材安装

安装控制板。将 Arduino UNO R3 放置在盒子内部，对应孔位，用微钻打孔，用螺丝将其固定在盒子底部。将 W5100 网络拓展板和 I/O 传感器扩展板（见图 15.2）按对应孔位，依次插在 Arduino UNO R3 控制板上。



■ 图 15.2 I/O 传感器扩展板

将红、黄、蓝三色 LED 发光模块、蜂鸣器，分别连接在扩展板数字口 6、7、8、4 上，模拟气体传感器连接在扩展板的模拟口 2 上，线路连接如图 15.3 所示，实物如图 15.4 所示。



■ 图 15.3 LED 发光模块、蜂鸣器连接示意图



■ 图 15.4 连接实物图

15.2 系统设置

15.2.1 登录平台

进入 Yeelink 平台注册用户，以注册用户身份登录平台，进入用户中心。如图 15.5 所示，登录后，单击头像可以进入“用户中心”。



■ 图 15.5 登录 Yeelink 平台

15.2.2 添加新设备

进入用户中心后，在页面左侧导航栏选择“我的设备”→“增加新设备”，如图

15.6 所示。同时右侧出现“增加新设备”页面。依次完成设备来源、设备名、标签 TAGS、描述、地理位置名称、地理位置数据等项目的设置。其中设备来源选择为自选设备，设备名为物联网测试项目，标签 TAGS 为烟雾，描述为烟雾，地理位置名称为你所在省市地区名称，地理位置数据请从地图放置一个标记，以表示设备所在的位置，坐标会被自动记录。以上设置完成后，单击“保存”按钮完成新设备的添加。



■ 图 15.6 添加新设备平台界面

15.2.3 添加传感器

单击 Yeelink 页面导航栏“我的设备”→“管理设备”，即可看到之前添加的设备（见图 15.7），选择需要添加传感器的设备，如刚才添加的“物联网测试项目”，进入该设备的界面，在页面左下方的传感器栏目下，单击“增加一个传感器”，即可为烟雾监测添加一个传感器。



■ 图 15.7 添加传感器界面

15.2.4 设置传感器

单击添加传感器后，进入传感器设置界面（见图 15.8）。设置传感器名为烟雾传感器，类型为数值型传感器，完成以上设置后，单击“保存”按钮。这样我们就建立了一个名为“物联网测试项目”的设备，同时该设备拥有了一个名为“烟雾传感器”的数值型传感器。



■ 图 15.8 设置传感器界面

如果需要查看设备编号和传感器编号，在上图中“URL”中 device 后面的数字“17617”为设备编号，sensor 后面的数字“30621”为烟雾传感器编号。

15.2.5 添加触发动作

选择“我的设备”→“管理动作”→“增加新动作”（见图 15.9），我们给动作起名为“烟雾报警”，类型选择为“微博发布”，

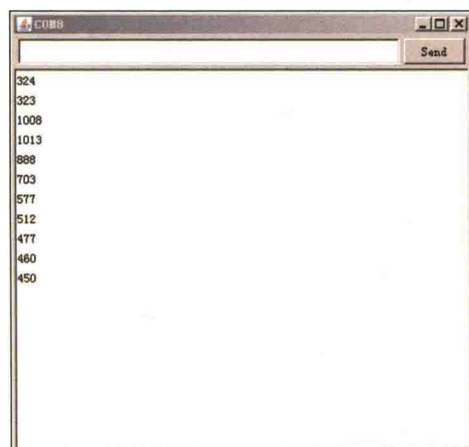
微博提供商为“新浪微博”，并获取授权。授权成功后，单击保存，此时我们已经建立了一个名为“烟雾报警”的动作。



■ 图 15.9 添加触发动作界面

添加完动作后，微博还不能进行播报，因为它不知道在什么情况下开始播报。而触发动作就是让微博明白在烟雾达到什么条件时发布微博。进入平台，单击“我的设备”，在“管理设备”中选择要播报的设备，选择“烟雾”，在其相应的传感器下方，单击“增加触发”为传感器增加一个触发动作。指定触发条件，这里我们选择烟雾 > 400 时触发。指定动作选项中，使用我们之前添

加的动作“微博播报”。消息内容中填写需要进行微博播报的内容，\$value 代表当前传感器的值。这样我们已经为烟雾传感器添加了一个触发动作，当烟雾 > 400 时，进行微博播报。这里好像有个问题，我们设置为 400，这个值是从哪里获得呢？其实这是通过串口监视器来获得传感器所采集到的值（见图 15.10）。



■ 图 15.10 传感器采集的值

15.3 代码编写

```
#include <Ethernet.h>
#include <WiFi.h>
#include <SPI.h>
#include <yl_data_point.h>
#include <yl_device.h>
#include <yl_w5100_client.h>
#include <yl_wifi_client.h>
#include <yl_messenger.h>
#include <yl_sensor.h>
#include <yl_value_data_point.h>
#include <yl_sensor.h>
//this example reads data from a lm35dz sensor, convert value to degree
Celsius
//and then post it to yeelink.net
//replace 2633 3539 with ur device id and sensor id
```

```

yl_device ardu(17617); // 此处替换为你的设备编号
//yl_sensor therm(26154, &ardu); // 此处替换为你的传感器编号
yl_sensor yw(30621, &ardu);
//replace first param value with ur u-apikey
yl_w5100_client client;
yl_messenger messenger(&client, «e1613c569602335b3d76c77cb5e29884»,
«api.yeelink.net»); // 此处替换为你自己的 API KEY
const int YW_PIN=A2; // 烟雾
float yw_convertor(int analog_num2)
{
    return analog_num2 * (5.0 / 1024.0 * 100);
}
void setup()
{
    Serial.begin(9600); //for output information
    pinMode(6,OUTPUT);
    pinMode(7,OUTPUT);
    pinMode(8,OUTPUT);
    pinMode(4,OUTPUT);
    byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xAA};
    Ethernet.begin(mac);
}
void loop()
{
    yl_sensor yw(30621, &ardu); // 此处替换为你的传感器编号 (烟雾)
    int y = analogRead(YW_PIN);
    Serial.println(y);
    if (y>900) {
        digitalWrite(6, HIGH);
        digitalWrite(7, LOW);
        digitalWrite(8, LOW);
        tone(4,100); // 发出指定频响
        pinMode(4,INPUT); // 关闭4号口输出,没有想到其他好办法,所以将状态改为输入
    }
    else
    {
        if (y>500)
        {
            digitalWrite(7, HIGH);
            digitalWrite(6, LOW);
            digitalWrite(8, LOW);
        }
        else
        {
            digitalWrite(8, HIGH);
            digitalWrite(6, LOW);
            digitalWrite(7, LOW);
        }
    }
}

```

```
}  
}  
yl_value_data_point dp2(yw_converter(y));  
yw.single_post(messenger, dp2);  
delay(1000);  
}
```

15.4 作品测试

输入以上代码后，点烟进行测试。经过设置，当传感器烟雾值 >400 时，Yeelink 就会自动发送微博，如图 15.11 所示。



图 15.11 测试界面

打开网络客户端，就可以观察烟雾的数据曲线了，如图 15.12 所示。



图 15.12 烟雾数据曲线

通过这个实例，相信你已经学会了如何通过微博播报传感器所捕捉到的数值了，接下来可以探索更多的动作和触发条件，实现更加丰富的功能。

16

DIY 空气净化器



针对雾霾，创客总得做点什么，至少先解决一下自己房间这种小范围的问题。于是我 DIY 了题图所示的 PM2.5 空气净化器。

虽然做得丑了点，但是效果还不错，房间里的 PM2.5 值从 70 多很快降低到 15 左右。

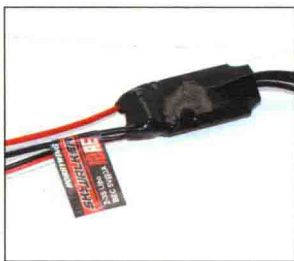
制作所需的器材见表 16.1。

表 16.1 项目器材

- 1** 涡轮增压三相无刷电机金属涵道



- 2** 好盈 40A 电子调速器（通常简称为“电调”）



- 3** 舵机测试器



- 4** 12V/30A 开关电源



- 5** 过滤网、滤芯



- 6** PLA 3D 打印耗材（直径 1.75mm，粉色、紫色、绿色）

- 7** 3D 打印机（笔者所用型号为 Overlord Pro）

- 8** PM2.5 激光粉尘传感器

- 9** Arduino UNO 控制板

- 10** LCD Keypad Shield 按键扩展板

- 11** 7.4V/2200mAh 锂电池（带充放电保护板）

- 12** 7.4V 锂电池充电器

- 13** 线材、螺丝、接头

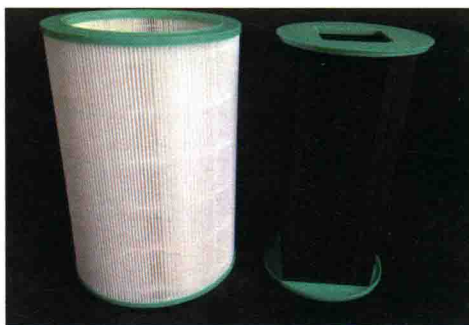
在开始制作之前，先让我们来了解一下空气净化器的原理。虽然市场上的空气净化器种类、名称、功能等不尽相同，但是从它的原理上来说，没有那么多种类。空气净化器从原理上来分，主要有两种，一种是被动吸附过滤式的，另一种则是主动式的。

被动式空气净化器主要是用风机将空气抽入机器，然后通过内置的滤网过滤空气，能够起到过滤粉尘、异味、消毒等作用。它们多是采用 HEPA 滤网 + 活性炭滤网 + 光触媒（冷触媒、多远触媒）+ 紫外线杀菌消毒 + 静电吸附滤网等方法来处理空气。而 HEPA 滤网有过滤粉尘颗粒物的作用，活性炭等物质主要是吸附有异味的有毒有害物质。

主动式空气净化器在原理上区别于被动式空气净化器就在于，它摆脱了风机与滤网的限制，不用被动地等到空气被抽送进来再进行过滤，然后再通过风机排出，而是有效、主动地向空气中释放净化、灭菌的因子，让它们在空气中弥漫、扩散，达到对室内各个角落的空气进行无死角净化的效果。

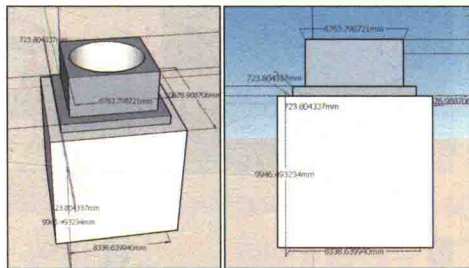
我们这里的 DIY 作品属于被动式空气净化器。

我们购买的滤网和滤芯如图 16.1 所示，内部带有活性炭。再结合购买的涵道，决定让涵道出风口安放在活性炭内侧。但是在实际安放过程中发现，两者结合不够紧密，且不好固定，而且涵道在高速运转时产生巨大的冲击力会使得两者分离。所以最好的解决方案就是自己来设计固定涵道和连接活性炭滤芯的风道。

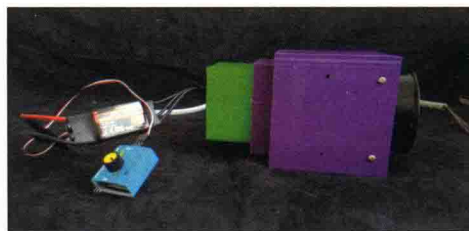


■ 图 16.1 滤网和滤芯

这里采用 Google SketchUp（谷歌草图）软件设计 3D 模型（见图 16.2，制作过程省略，可参考前面的《3D 建模 + 打印个性化钥匙扣》），使用 3D 打印机打印，实际效果如图 16.3 所示。这里有个小插曲，设计 3D 模型时忘记画螺丝孔，等打印时才发现，只好再来钻孔。



■ 图 16.2 风道的 3D 模型



■ 图 16.3 风道的实际打印效果（采用双色打印完成）

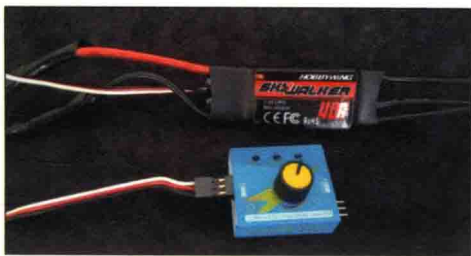
电调的 3 根输出线和涵道电机的 3 根线可以任意连接（见图 16.4），当发现风向

不是自己所需要的时，调换任意两根线即可。

将电调的信号线插在舵机测试仪上（见图 16.5），请注意，舵机测试仪左边有一排针，右边有三排针，一排针的这端是电源输入端。因为电调本身就输出给舵机测试仪，所以电调的信号线只需要插三排针的任意一排即可，请注意，插头露出金属的部分朝上。连接好后，将电调穿过风道，使得涵道电机进入固定槽位（见图 16.6），并用螺丝固定。



■ 图 16.4 连接电调和涵道电机



■ 图 16.5 连接电调和舵机测试仪



■ 图 16.6 让涵道电机进入风道的固定槽位内

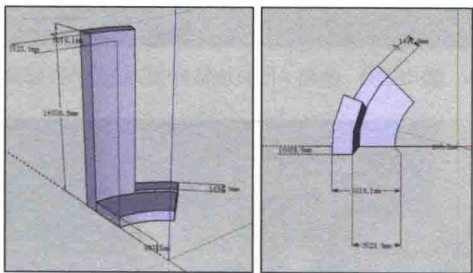
电调的电源线接 12V 电池或电源（见图 16.7），正负极绝不能接错，否则可能会烧毁。



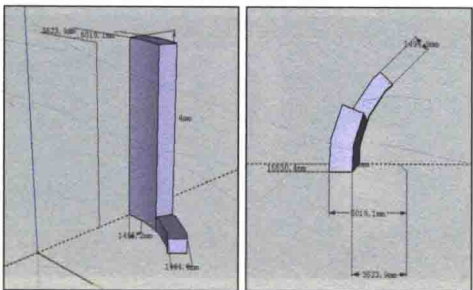
■ 图 16.7 连接电调与电源

全部连接好后，请保证舵机测试仪油门在最小，然后接通电源，等测试仪蓝灯常亮，就可以旋转油门进行操作了。

完成上面步骤后，虽然实现了除 PM2.5 的功能，但是作为一个要求“颜值”的创客，我们还需要一个外壳，于是又有了图 16.8、图 16.9 所示的 3D 模型文件。



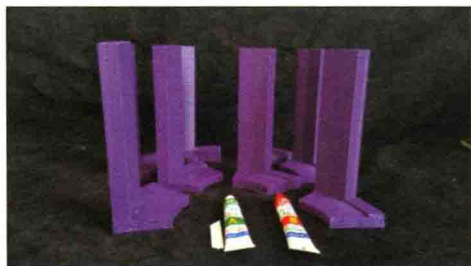
■ 图 16.8 外壳下半部分模型文件（打印 8 块，紫色）



■ 图 16.9 外壳上半部分模型文件（打印 8 块，粉色）

将图 16.8、图 16.9 所示的部件打

印好以后，使用 AB 胶水进行粘合（见图 16.10、图 16.11）。在图 16.11 中，可以看到风道和滤网、滤芯的连接方式。

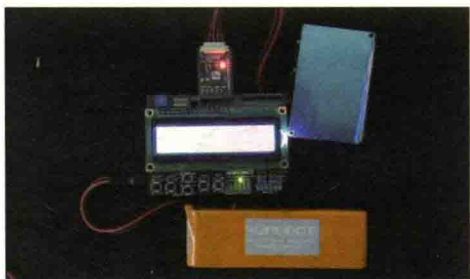


■ 图 16.10 使用 AB 胶水粘合 3D 打印外壳部件



■ 图 16.11 拼合外壳

空气净化器做好了，还需要检验净化的效果，这里使用了 DF 的激光 PM2.5 传感器（见图 16.12）。其使用方法在官方网站上有详细介绍，这里不再重复，请自行查阅以下地址：[http://wiki.dfrobot.com.cn/index.php/\(SKU:SEN0177\)PM2.5激光粉尘传感器](http://wiki.dfrobot.com.cn/index.php/(SKU:SEN0177)PM2.5激光粉尘传感器)。



■ 图 16.12 使用激光 PM2.5 传感器检验净化效果

你千万千万别以为我们的 PM2.5 空气净化器这就完成了，如果这样，不光对不起各位观众，也对不起自己。目前的这个净化器是手动的，能不能改成自动的？能不能有物联网功能？能不能……脑洞大开的创客们，行动起来吧！

无线电 出品



Arduino+3D 打印 创新电子制作

16 个智能硬件搭建实例

3D 打印为创造带来了全新的工具，是人类生产资料的又一次升级。陈杰老师的创意电子制作为我们淋漓尽致地展现了如何将 3D 打印和开源硬件结合，并创造出令人耳目一新的项目，一扇全新的创造之门由此徐徐展开。

——DFRobot 创始人 /CEO 叶琛

创客分享智慧，设计改变世界。本书以开源的 Arduino 智能控制、富有创意的 3D 打印技术，为你带来技术“造物”之趣，点燃技术创新之火！

——教育部普通高中通用技术课程标准修订组核心成员、
江苏省教育科学研究院副研究员 陈向阳



封面设计 / 周桂红

DF 创客社区网址: www.DFRobot.com.cn



扫描二维码
带你轻松玩转开源项目

分类建议: 电子技术 / 制作项目

人民邮电出版社网址: www.ptpress.com.cn

ISBN 978-7-115-43213-1



ISBN 978-7-115-43213-1

定价: 39.00 元