

机器人综合基础实践教程

(入门篇、提高篇)

目录

入门篇	1
引言	2
机械零件简介	3
实验 1 造型设计	6
主控板简介	10
编程环境	19
实验 2 blink	21
扩展板简介	22
实验 3 驱动轮模块	25
实验 4 双轮万向车	29
传感器简介	32
实验 5 触碰触须启动小车	38
实验 6 红外启动小车	41
实验 7 带传动模块	42
实验 8 履带机器人	44
实验 9 红绿灯	46
实验 10 声控灯	48
实验 11 光感小夜灯	51
实验 12 红外迎宾装置	54
实验 13 机器人简单循迹	56
实验 14 机器人躲避悬崖	61

实验 15 机器人悬崖漫步	62
提高篇	63
实验 16 关节模块	64
实验 17 2 自由度云台	68
实验 18 数字量传感器测值	70
实验 19 模拟量传感器测值	72
实验 20 超声波传感器测距算法	75
实验 21 超声雷达实验	80
实验 22 机械手爪模块	82
实验 23 3 自由度机械臂	85
实验 24 按颜色分拣	86
实验 25 有限状态机算法实验	87
实验 26 LED 点阵笑脸	99
实验 27 蓝牙电子温度计实验	104
实验 28 速度协调实验	107
实验 29 语音识别实验	110
实验 30 双轮小车平衡实验	113
实验 31 蓝牙排爆机器人	117
实验 32 WiFi 视频排爆	121

入门篇

引言

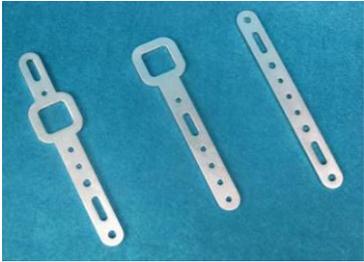
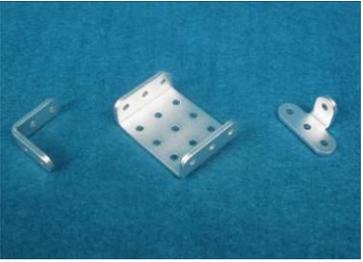
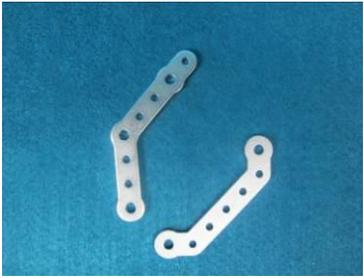
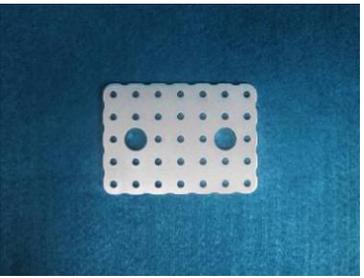
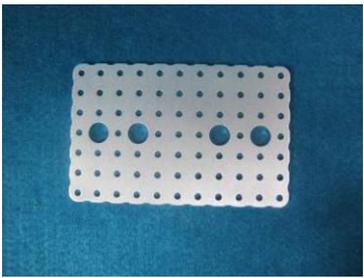
从“大众创业 万众创新”创客运动所带来的人才和项目创新方面的思考，基于高校“人才的输出中心”这个核心功能定位，应该担当的位置是工程创新人才的培育中心，是支持当下社会创新产业升级，以及创新产业创业的创新人才能量供应中心。

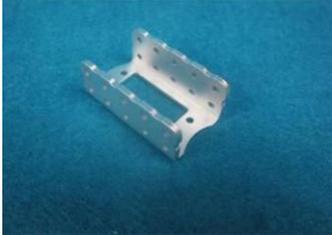
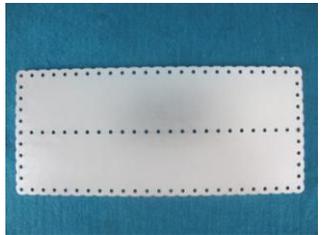
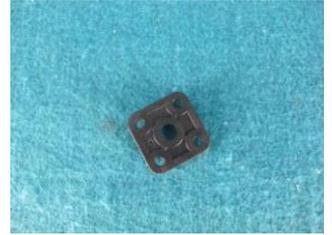
国务院副总理刘延东出席深入推进高校创新创业教育改革座谈会中强调，人是创新最关键因素，创新驱动是人才驱动。要全面落实党中央、国务院决策部署，认真贯彻李克强总理重要批示精神，切实增强深入推进高校创新创业教育改革的责任感和紧迫感，全面提高人才培养质量，为促进大众创业万众创新和建设创新型国家提供有力人才支撑。

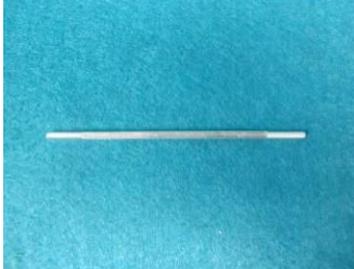
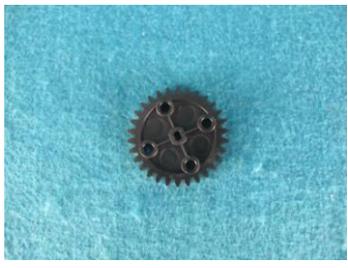
正是在当下国家产业发展规划背景下，传统高校工程教育已经难以满足创新人才政策的培养需求，且工业 4.0 时代下，如“创客 Maker”、“机器人”、“模块化”、“智能制造工厂”、“智能硬件”、“创新创业”、“互联网信息化”、“校企拟合”等新兴理念大量涌入高校工程教育改革当中，创新工程教育以及创新人才培养，不只是单一学科单一方向的教学改革，而是涵盖多方面多角度的综合应用型创新体系，不只是传统高校教学实践内容，还包含了企业真实环境下的创新设计思维教育等等，所以，如何落实当下创新工程教育，以及在中国制造 2025 规划下，创新应用型人才的培养，是当下高校工程教育改革的难题。

机械零件简介

(下表仅包含常见零件, 不同产品所配零件不同, 请以配置清单为准)

		
双足腿、四足连杆、双足支杆	90度支架、3×5折弯、输出支架	机械手 20mm、机械手 40mm、机械手 40mm 驱动
曲柄滑块机构的主要零件, 可用于搭建机器人行走机构。	三种折弯件, 可搭建机构支架, 连接不同平面	三种长短不同的连杆, 可搭建四连杆结构, 伸缩机械手等
		
机械手指、双足连杆	垫片 10、垫片 20、轮支架、10mm 滑轨	5×7 孔平板
两种带角度的连杆, 可搭建机械手爪, 腿部结构等。	四种小金属件, 主要起调节机构层次的作用	长边 7 个孔, 短边 5 个孔, 可用作小型搭载平台
		
7×11 孔平板	小轮	舵机双折弯
长边 11 个孔, 短边 7 个孔, 可用作大型搭载平台	可用作履带、滚筒的骨架	可用作机器人关节摆动部件

		
大轮	马达支架	大舵机支架
可用做大轮子，机架、半球结构、球结构等	小型舵机使用，连接小型舵机与其他零件	连接大舵机与其他零件
		
大舵机 U 型支架	牛眼万向轮	11×25 孔平板
用于大舵机组装关节式结构	国际标准零件	可用作大型机架平台
		
输出头	马达后盖输出头	大舵机输出头
		
大舵机后盖输出头	直流马达输出头	履带片

		
<p>三种球形件</p>	<p>直流马达支架</p>	<p>传动轴</p>
<p>可用于翅膀、腿、轮足等仿生机构的搭建</p>	<p>直流马达的支架，连接直流电机与其他零件</p>	<p>不锈钢传动部件，可连接齿轮等，两端是扁的</p>
		
<p>双足大腿、双足小腿</p>	<p>双足脚</p>	<p>两种偏心轮</p>
<p>可组装特殊的曲柄滑块，用于机器人行走机构。</p>	<p>可作为脚使用，也可用于其它功能</p>	<p>可组装偏心轮机构，代替凸轮，代替曲柄等</p>
		
<p>30 齿齿轮</p>	<p>随动齿轮</p>	<p>联轴器、1:10 模型轮胎</p>
		
<p>不锈钢轴套 轴套 2.7、轴套 5.4、轴套 10.4、轴套 15.4</p>	<p>国际标准尼龙螺柱 螺柱 10、螺柱 15、螺柱 20、螺柱 30；35mm 金属螺柱</p>	<p>国际标准 M3 不锈钢螺丝、螺母</p>

实验 1 造型设计

实验目的：1、锻炼空间构型能力；2、锻炼联想力；2、感性理解探索者的零件体系；3、熟悉探索者零件，能够根据不同情况灵活使用零件；4、练习使用组装工具；

实验性质：验证型实验、设计型实验

实验课时：1 课时

实验内容：

1. 零件的固定

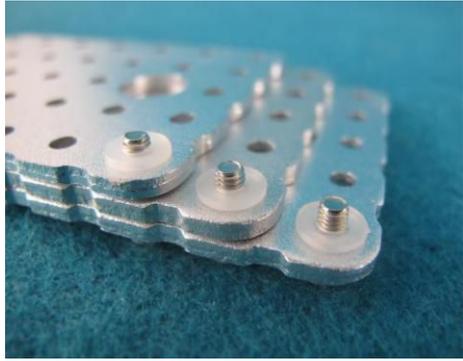
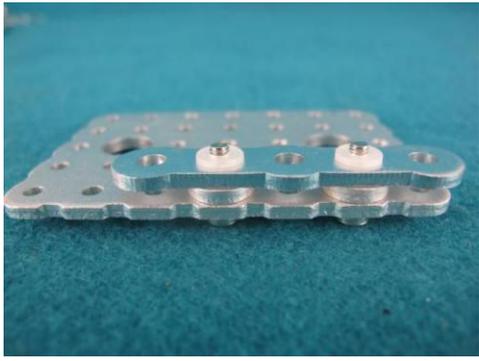
固定连接的零件紧紧连在一起，不能移动，也不能转动。根据几何学原理可知，两点决定一条直线，因此要固定两个零件，至少需要用到两颗螺丝。



“探索者”零件上的 3mm 孔通常用于零件的固定。固定时，有时候会用到垫片、螺柱等。

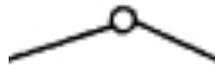
请完成下表中的六种连接，用螺丝配合螺母或防滑垫连接。

	
连杆零件的固定	平板零件的固定

	
螺柱参与连接	多层次连接
	
垫片参与连接	多层次螺柱固定

2. 零件的铰接

铰接是指连接在一起的零件不能移动,但可以转动,铰接的长杆型零件只有一个连接点。日常所见如折叠刀、圆规、折尺、以及从动轮等。



4mm 孔加入轴套可以实现铰接功能。

请你参考下表,找到轴套,完成表中的 5 种铰接。

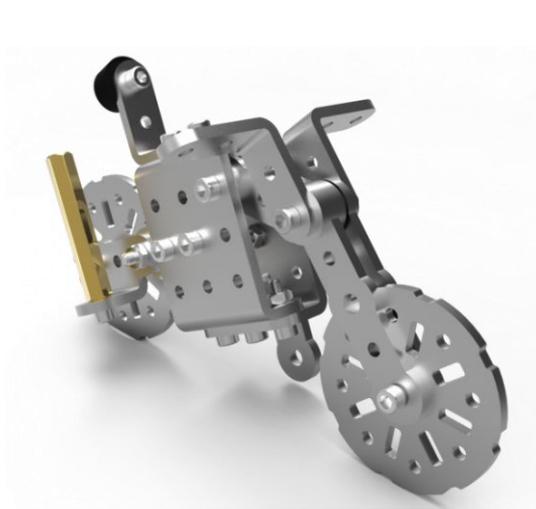
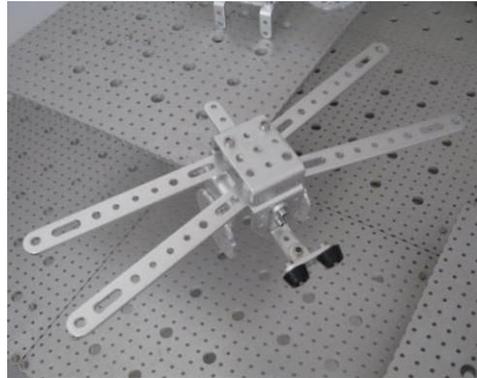
	
轴套	一个小型铰接结构

	
多个零件铰接	多层次铰接
	
螺柱参与铰接	滑槽参与铰接

3. 请从以下题目中选择一些进行搭建，你可以参考图片，也可以自己发挥，搭建简单的积木造型，不用带电机。

- (1) 家具造型：桌、椅、板、凳、柜子、带合页的门、安全门等；
- (2) 几何造型：立方体、棱柱、圆柱、梯形体、雪花体、八边形体等；
- (3) 轮类造型：滑板、轮滑鞋、自行车、汽车、飞机、拖拉机、挖掘机等；
- (4) 机械造型：等臂四杆、不等臂四杆、六杆、摆角放大机构、曲柄滑块等；
- (5) 工具造型：手机支架、开瓶器、书本支架、笔筒、抽屉锁、梯子等；
- (6) 武器造型：刀枪剑戟斧钺勾叉等；枪械、坦克、火炮等；
- (7) 动物造型：狗、鸟、恐龙、鱼、蛇、昆虫等；
- (8) 建筑造型：平房、楼房、桥梁、风车等。

示例：



主控板简介

1. Basra 主控板 (本教材大部分实验例程基于 Basra 编写)

概述

Basra 是一款基于 Arduino 开源方案设计的一款开发板，通过各种各样的传感器来感知环境，通过控制灯光、马达和其他的装置来反馈、影响环境。板子上的微控制器可以在 Arduino、eclipse、Visual Studio 等 IDE 中通过 c/c++ 语言来编写程序，编译成二进制文件，烧录进微控制器。Basra 的处理器核心是 ATmega328，同时具有 14 路数字输入/输出口 (其中 6 路可作为 PWM 输出)，6 路模拟输入，一个 16MHz 晶体振荡器，一个 USB 口，一个电源插座，一个 ICSP header 和一个复位按钮。

主 CPU 采用 AVR ATMEGA328 型控制芯片，支持 C 语言编程方式；该系统的硬件电路包括：电源电路、串口通信电路、MCU 基本电路、烧写接口、显示模块、AD/DA 转换模块、输入模块、IIC 存储模块等其他电路模块电路。控制板尺寸不超过 60*60mm，便于安装。CPU 硬件软件全部开放，除能完成对小车控制外，还能使用本实验板完成单片机所有基础实验。供电范围宽泛，支持 5v~9v 的电压，干电池或锂电池都适用。编程器集成在控制板上，通过 USB 大小口的方式与电脑连接。下载程序。开放全部底层源代码。控制板含 3A6V 的稳压芯片，可为舵机提供 6v 额定电压。板载 8*8led 模块采用 MAX7219 驱动芯片。板载一片直流电机驱动芯片 FAN8100MTC，可同时驱动两个直流电机。板载 USB 驱动芯片及自动复位电路，烧录程序时无需手动复位。2 个 2*5 的杜邦座扩展坞，方便无线模块、OLED、蓝牙等扩展模块直插连接，无需额外接线。

特点

- 开放源代码的电路图设计，程序开发接口免费下载，也可依需求自己修改。
- 可以采用 USB 接口供电，不需外接电源，也可以使用外部 DC 输入。
- 支持 ISP 在线烧，可以将新的“bootloader”固件烧入芯片。有了 bootloader 之后，可以在线更新固件。
- 支持多种互动程序，如：Flash、Max/Msp、VWV、PD、C、Processing 等。
- 具有宽泛的供电范围，电源电压可任选 3v~12v 的电源
- 采用堆叠设计，可任意扩展
- 主控板尺寸不超过 60mm*60mm，便于给小型机电设备安装
- 板载 USB 驱动芯片及自动复位电路，烧录程序时无需手动复位

参数

- 处理器 ATmega328
- 工作电压 5V
- 输入电压 (推荐) 7-12V
- 输入电压 (范围) 6-20V
- 数字 IO 脚 14 (其中 6 路作为 PWM 输出)
- 模拟输入脚 6
- IO 脚直流电流 40 mA
- 3.3V 脚直流电流 50 mA
- Flash Memory 32 KB (ATmega328 , 其中 0.5 KB 用于 bootloader)
- SRAM 2 KB (ATmega328)
- EEPROM 1 KB (ATmega328)
- 工作时钟 16 MHz

电源

Basra 可以通过 3 种方式供电, 而且能自动选择供电方式

- 外部直流电源通过电源插座供电。
- 电池连接电源连接器的 GND 和 VIN 引脚。
- USB 接口直接供电。

电源引脚说明

VIN --- 当外部直流电源接入电源插座时, 可以通过 VIN 向外部供电; 也可以通过此引脚向 UNO 直接供电; VIN 有电时将忽略从 USB 或者其他引脚接入的电源。

5V --- 通过稳压器或 USB 的 5V 电压, 为 UNO 上的 5V 芯片供电。

3.3V --- 通过稳压器产生的 3.3V 电压, 最大驱动电流 50mA。

GND --- 地脚。

存储器

ATmega328 包括了片上 32KB Flash, 其中 0.5KB 用于 Bootloader。同时还有 2KB SRAM 和 1KB EEPROM。

输入输出

14 路数字输入输出: 工作电压为 5V, 每一路能输出和接入最大电流为 40mA。每一路配置了 20-50K 欧姆内部上拉电阻 (默认不连接)。除此之外, 有些引脚有特定的功能

- 串口信号 RX (0 号) TX (1 号): 与内部 ATmega8U2 USB-to-TTL 芯片相连,

提供 TTL 电压水平的串口接收信号。

- 外部中断 (2 号和 3 号): 触发中断引脚, 可设成上升沿、下降沿或同时触发。
- 脉冲宽度调制 PWM (3、5、6、9、10、11): 提供 6 路 8 位 PWM 输出。
- SPI (10(SS), 11(MOSI), 12(MISO), 13(SCK)): SPI 通信接口。
- LED (13 号): Arduino 专门用于测试 LED 的保留接口, 输出为高时点亮 LED, 反之输出为低时 LED 熄灭。

6 路模拟输入 A0 到 A5 : 每一路具有 10 位的分辨率 (即输入有 1024 个不同值), 默认输入信号范围为 0 到 5V, 可以通过 AREF 调整输入上限。除此之外, 有些引脚有特定功能

- TWI 接口 (SDA A4 和 SCL A5): 支持通信接口 (兼容 I2C 总线)。

AREF : 模拟输入信号的参考电压。

Reset : 信号为低时复位单片机芯片。

通信

- 串口 : ATmega328 内置的 UART 可以通过数字口 0 (RX) 和 1 (TX) 与外部实现串口通信 ; ATmega16U2 可以访问数字口实现 USB 上的虚拟串口。
- TWI (兼容 I2C) 接口
- SPI 接口

下载程序

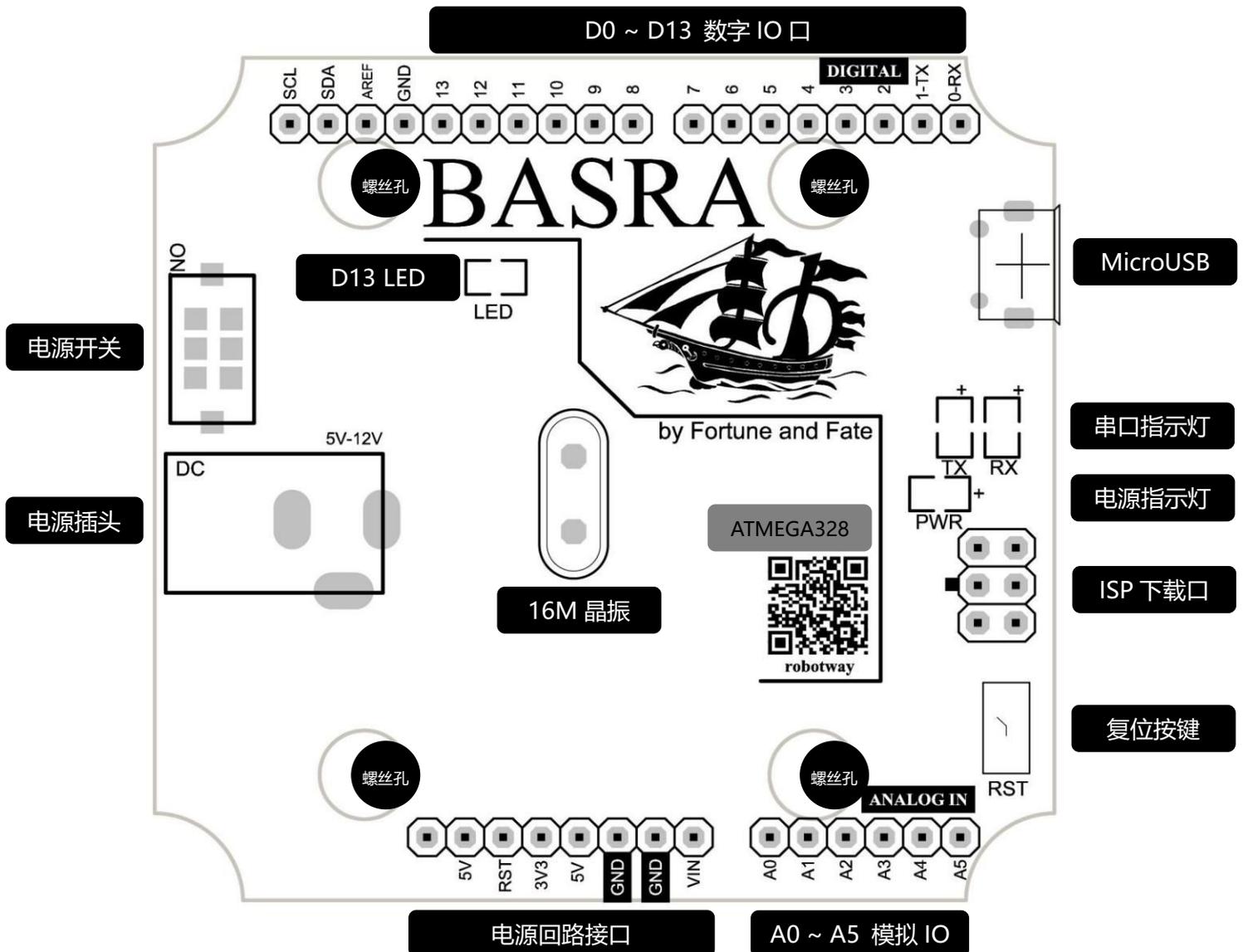
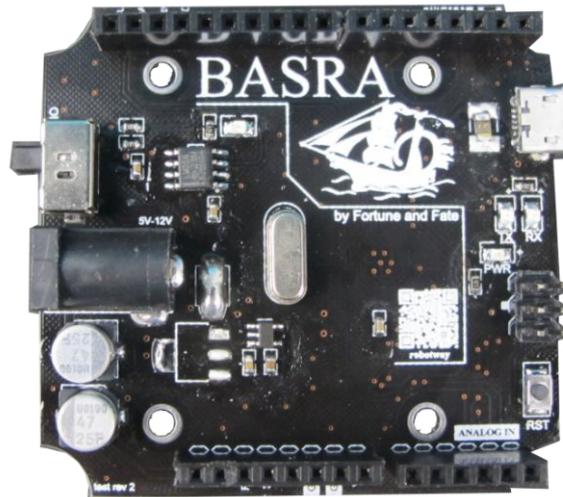
Basra 上的 ATmega328 已经预置了 bootloader 程序, 因此可以通过 Arduino 软件直接下载程序到主控板中。

可以直接通过主控板上 ICSP header 直接下载程序到 ATmega328。

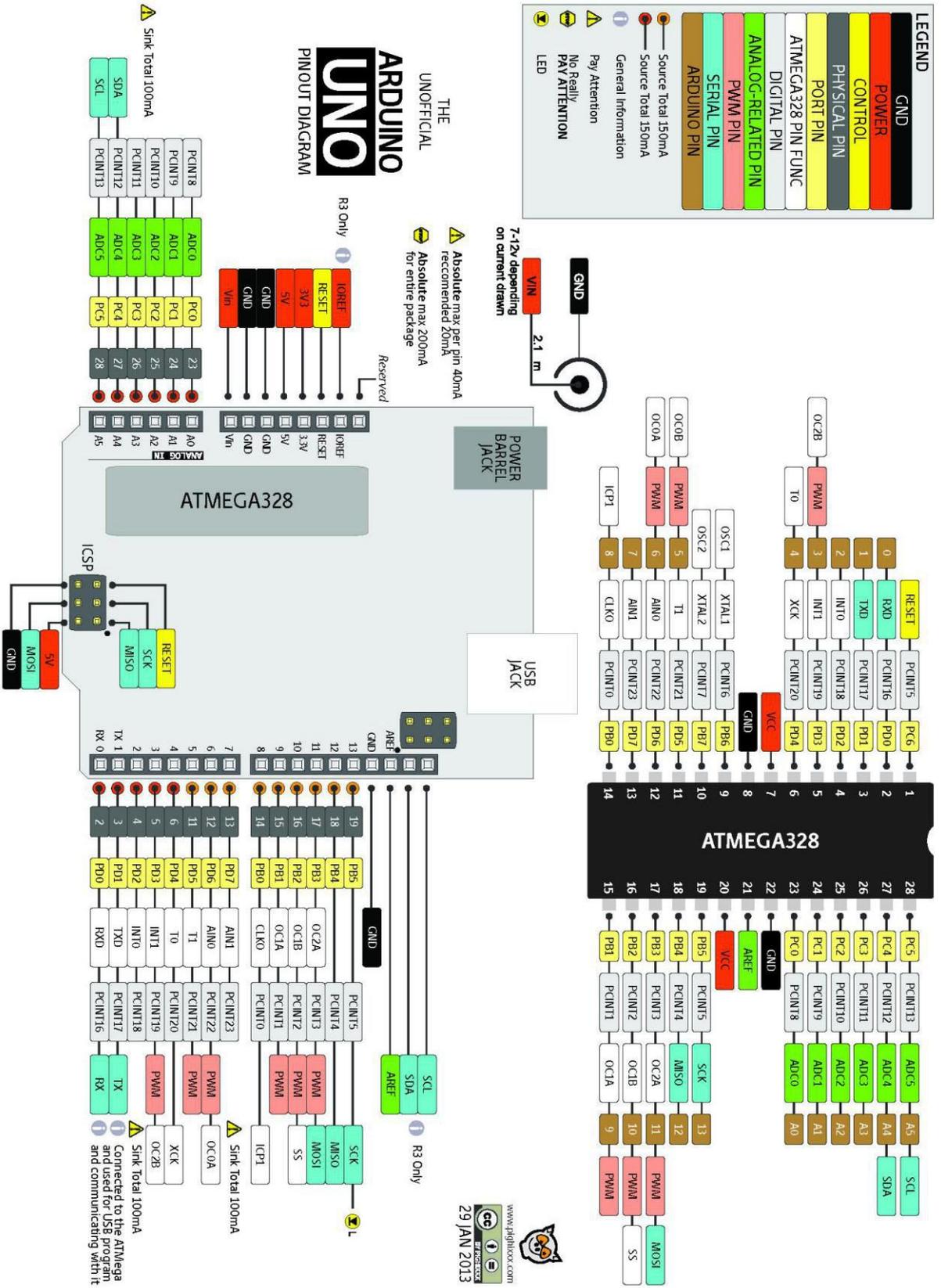
注意事项

- USB 口附近有一个可重置的保险丝, 对电路起到保护作用。当电流超过 500mA 时会断开 USB 连接。
- 主控板提供了自动复位设计, 可以通过主机复位。这样通过 Arduino 软件下载程序到主控板中时, 软件可以自动复位, 不需要在复位按钮。

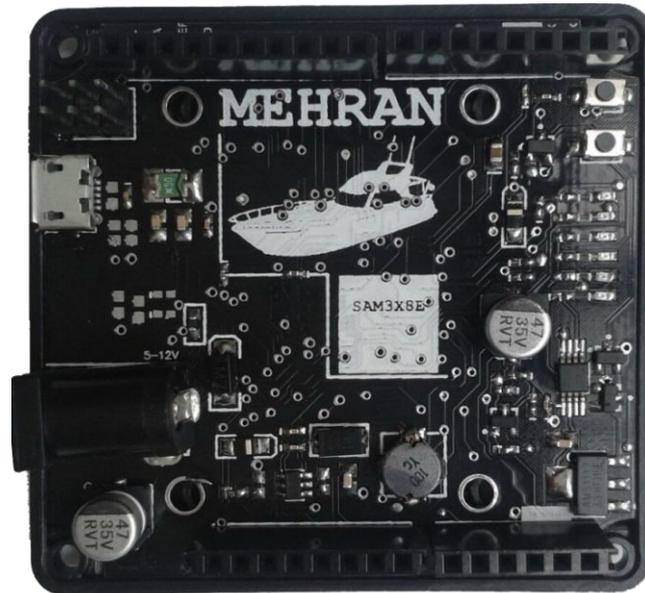
实物图片与接口



芯片与接口引脚



2. Mehran 控制板



概述

Mehran 是一块基于 Atmel SAM3X8E CPU 的微控制器板。它有 14 个数字 IO 口 (其中 12 个可用于 PWM 输出), 6 个模拟输入口, 1 路 UART 硬件串口, 84 MHz 的时钟频率, 一个 USB OTG 接口, 一个电源插座, 一个复位按键和一个擦写按键。

电路板上已经包含控制运行所需的各种部件, 你仅需要通过 USB 连接到电脑 或者通过 AC-DC 适配器、电池连接到电源插座就可以让控制器开始运行。Mehran 兼容工作在 3.3V 且引脚排列符合 arduino 标准的 arduino 扩展板。

特点

- 开放源代码的电路图设计, 程序开发接口免费下载, 也可依需求自己修改。
- 可以采用 USB 接口供电, 不需外接电源, 也可以使用外部 DC 输入。
- 支持 ISP 在线烧, 可以将新的 “bootloader” 固件烧入芯片。有了 bootloader 之后, 可以在线更新固件。
- 支持多种互动程序, 如: Flash、Max/Msp、VWV、PD、C、Processing 等。
- 具有宽泛的供电范围, 电源电压可任选 3v~12v 的电源
- 采用堆叠设计, 可任意扩展
- 主控板尺寸不超过 60mm*60mm, 便于给小型机电设备安装
- 板载 USB 驱动芯片及自动复位电路, 烧录程序时无需手动复位

- 支持 Visual Studio、eclipse、sublime text 等 IDE

参数

- 微控制器 AT91SAM3X8E
- 工作电压 Voltage 3.3V
- 输入电压 (推荐) 7-12V
- 输入电压 (限制) 6-20V
- 数字 I/O 引脚 14(其中 12 路 PWM 输出)
- 模拟输入通道 6
- IO 口总输出电流 130 mA
- 3.3V 端口输出能力 800 mA
- 5V 端口输出能力 800 mA
- Flash 512 KB (所有空间都可以储存用户程序)
- SRAM 96 KB (两个部分: 64KB and 32KB)
- 时钟速率 84 MHz

电源

- Mehran 可以通过 Micro USB 接口或外接电源供电，电源可以自动被选择。
- 外部 (非 USB) 电源可以用 AC-DC 适配器 (wall-wart) 或电池。适配器可以插在一个 2.1 毫米规格中心是正极的电源插座上，以此连接到控制器电源。控制器能支持 6 到 20V 电压输入。如果输入电压低于 7V, 5V 引脚可能得到的电压会低于 5V, 控制器运行可能会不稳定。如果使用输入电压超过 12V, 可能会造成控制器过热，甚至损坏损坏。推荐输入电压范围为 7 到 12V。电源相关引脚如下：
- VIN. Mehran 使用外部电源时的输入电压引脚(你可以通过这个引脚供电，或者当 DC 座供电时，通过这个引脚使用 DC 电源)
- 5V. 通过板载稳压芯片输出的 5V 的电压引脚。你可以从 DC 电源口、USB、VIN 三处给控制器供电。也可以绕过稳压 IC，直接从 5V、3V3 引脚供电，但不建议这样使用。
- 3.3V. 通过板载稳压芯片输出的 3.3V 的电压引脚。最大电流 800 mA。该电压也是 SAM3X 的工作电压。
- GND. 接地引脚。

存储空间

SAM3X 有 512 KB (两块 256 KB)用于存储用户程序的 Flash 空间。ATMEL 已经在生

产芯片时，将 bootloader 预烧写进了 ROM 里。SRAM 有 96 KB，由两个连续空间 64 KB 和 32 KB 组成。所有可用存储空间 (Flash, RAM 和 ROM) 都可以直接寻址。你可以通过板子上的擦写按钮 擦除 SAM3X 的 Flash 中的数据。这个操作将删除当前加载的项目。在通电状态下，按住擦写按钮几秒钟，即可擦写。

输入和输出

- Digital I/O: 0 至 13 号引脚
- PWM: 引脚 2 to 13
- "L" LED: 13
- TWI 1: 20 (SDA) 和 21 (SCL)
- 支持使用 Wire 库来进行 TWI 通信
- 模拟输入: A0 到 A5
- Reset

通信

Mehran 可以通过多种方式与电脑、其他 arduino 或者其他控制器通信，也可以和其他不同的设备通信，像手机、平板、相机等等。SAM3X 提供一组硬件 UART 来进行串行通信。

原生的 USB 口虚拟串行 CDC 通信，这样可以提供一个串口，与串口监视器 或者 你电脑上其他应用相连。这个 USB 口也可以用来模拟一个 USB 鼠标或者键盘。要使用这个功能，请查看鼠标键盘库支持页面。这个原生 USB 口也可以作为 USB 主机去连接其他外设，如鼠标、键盘、智能手机。

编程

Mehran 通过 ArduinoIDE 中的 "download" 下载程序。在 SAM3X 的 arduino 上 上传程序和 AVR 控制器有所不同，这是因为 flash 在上传程序前，需要被擦写。SAM3X 的 ROM 中的程序会进行上传任务，但运行该程序的前提是 SAM3X 的 flash 空间是空的。

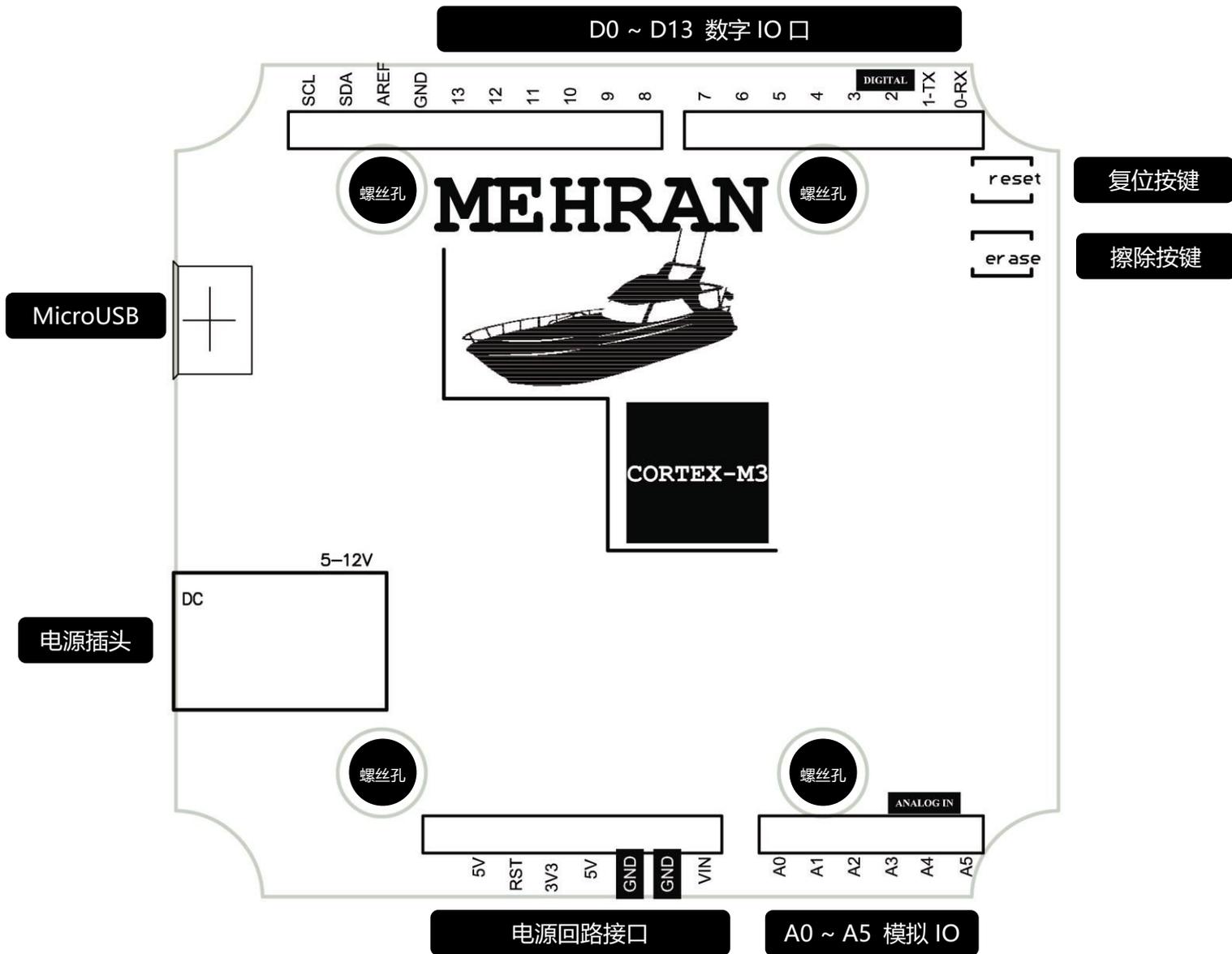
你需要在 arduinoIDE 中选择 "Arduino Due (Native USB Port)" 作为你的板子。连接原生 USB 端口 (靠近复位按键的那一个) 到你的电脑。在 1200 波特率下，打开和关闭串口会触发 SAM3X 的软擦写程序：flash 空间被擦写、程序倒转到 bootloader 区。如果主芯片损坏，软擦写程序 会不工作，这是因为该程序完全在 SAM3X 上。开关原生 USB 端口 baudrate 不会复位 SAM3X。

不同于其他的 Arduino 控制器使用 avrdude 上传程序，Mehran 上传程序依赖于 bossac。

USB 过流保护

Mehran 上有一个自恢复保险丝，短路或者过流时，可以自动断开，从而保护你的电脑 USB。大部分电脑都带有内部过流保护，这个保险丝可以提供一层额外保护。当电流大于 500MA 时，这个保险丝会自动断开，直到没有过载或者短路现象。

接口

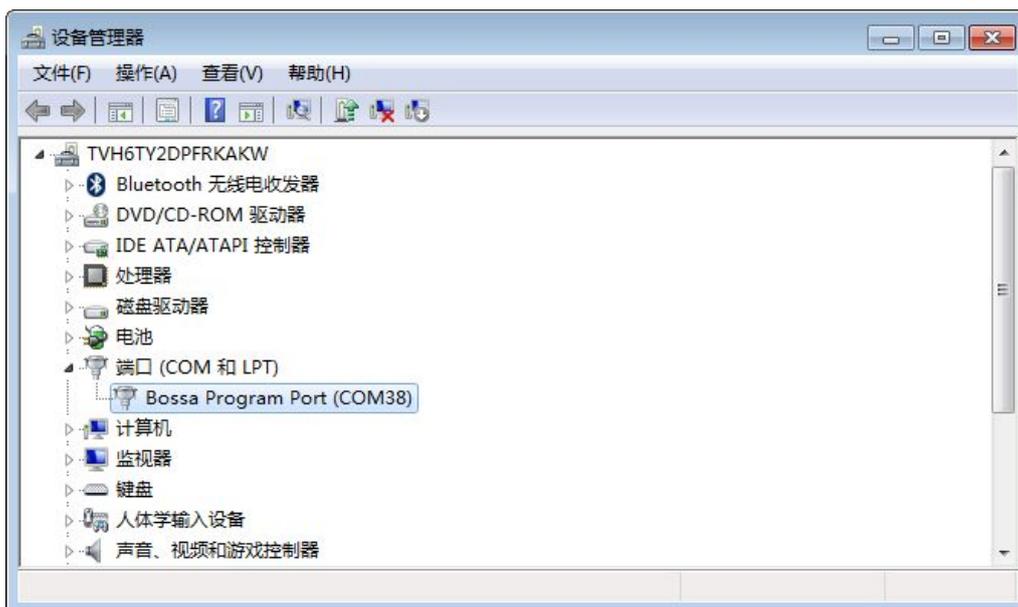


编程环境

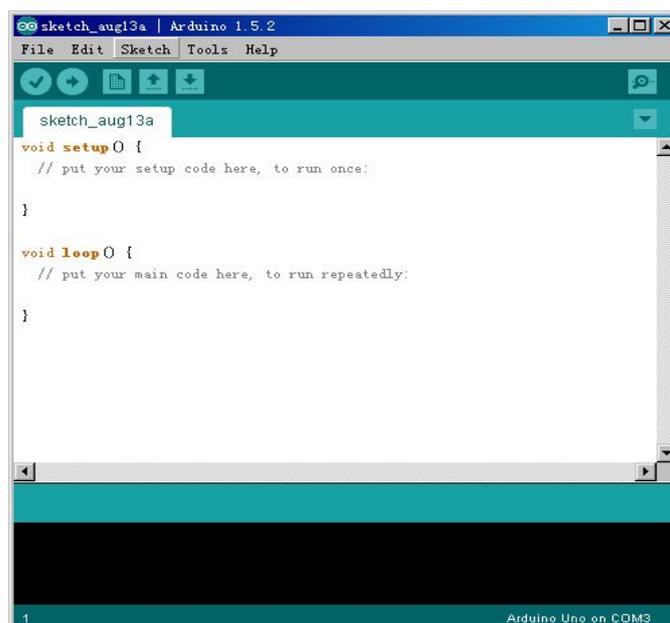
步骤一：拷贝..\tools\arduino-1.5.2 目录至本机位置。

步骤二：将 Mehran 控制板通过 miniUSB 数据线与 PC 连接，初次连接时会弹出驱动安装提示。选择..\tools\arduino-1.5.2\drivers\目录安装驱动。

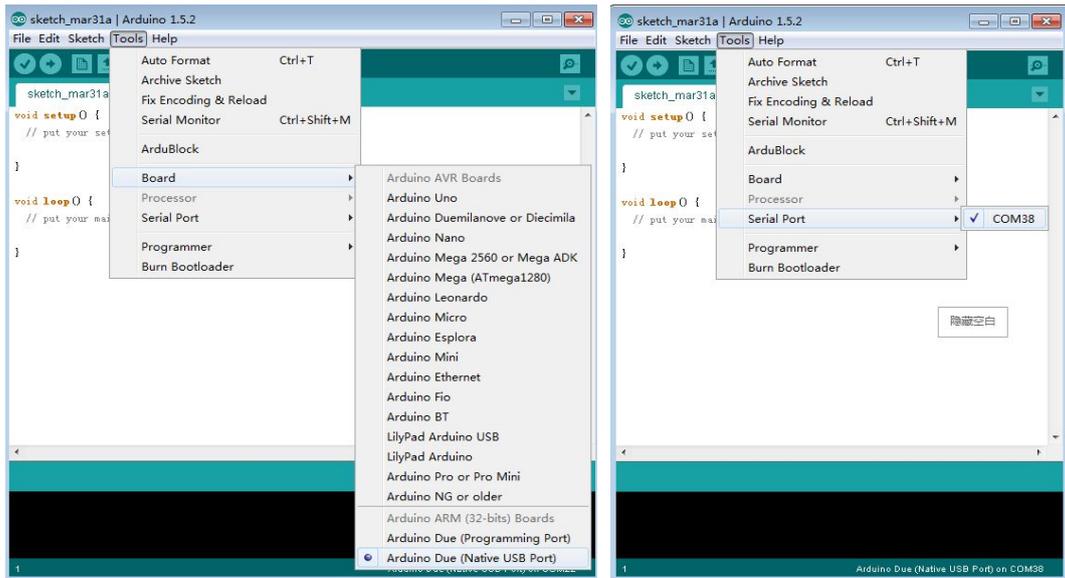
步骤三：打开设备管理器，在端口列表中，出现 Bossa Program Port(COMx)，表示驱动安装成功。请记录下这个 COM 端口号 x，下图中端口号为 COM38。



步骤四：在本机上运行 arduino-1.5.2 目录下的 arduino.exe，显示如下界面：

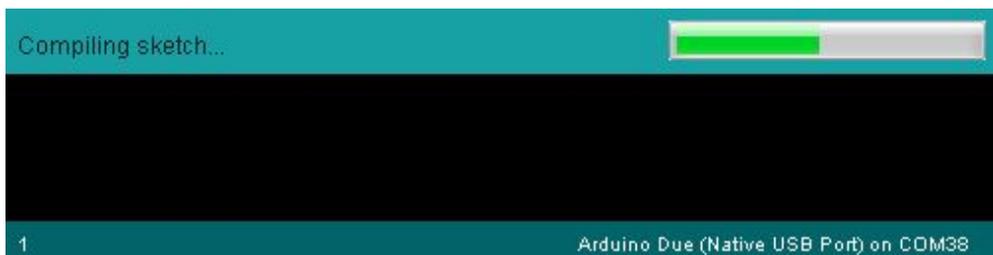


步骤五：在 Tools 菜单下，依次选择 Board 里的 Arduino Due(Native USB Port) 项，以及 Serial Port 里的 COM38 (COM38 为步骤 3 里记录下的端口号)。

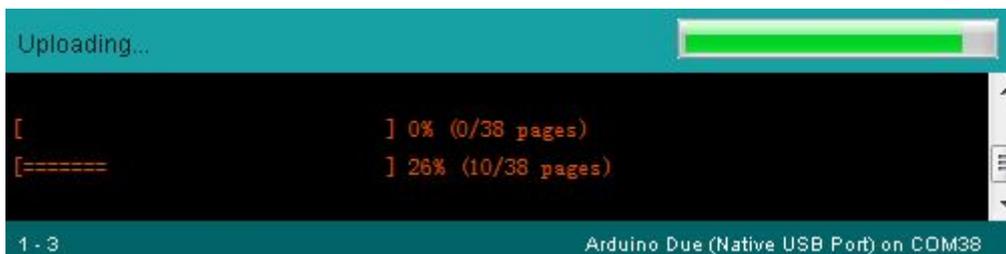


步骤六：点击 upload 按钮 ，一个空白的程序将自动烧录进 Mehran 控制板。

具体过程如下图所示：



1、开始编译代码



2、开始向 Mehran 控制板烧录程序，烧录过程中控制板上的 TX/RX 指示灯闪动



3、烧录成功

实验 2 blink

实验目的：学习使用延时语句，理解程序的顺序执行

实验性质：验证型实验、设计型实验

实验课时：1 课时

实验内容：

1. 13 号端口 LED 灯红色闪烁

从图形化界面左侧菜单栏拖选语句图形至右侧，完成下图所示语句



烧录，观察这段程序实现的效果是什么。

在这段程序中，我们使用了“延迟”语句，设定的参数是 1000，也就是 1000 毫秒，或者说 1 秒。“延迟 1000 毫秒”的意思**不是**说“1 秒之后再做”，而是说该语句上面所设定的状态，即“引脚 13 置高”，要保持 1 秒。这样，我们就能理解这段程序的意思了。

和传统的 C 语言一样，程序是从上到下顺序运行的，然后不断地循环。

2. 接下来请大家实现改变闪烁的频率。

扩展板简介

通过 BigFish 扩展板连接的电路可靠稳定，上面还扩展了伺服电机接口、8*8Led 点阵、直流电机驱动以及一个通用扩展接口，可以说控制板的必备配件。

特点

- (2) 完全兼容 Basra、Mehran 控制板接口
- (3) 彩色分组插针，一目了然
- (4) 全部镀金插针，用料考究，电器性能稳定
- (5) 优秀 PCB 设计，美观大方
- (6) 多种特殊接口设计，兼容所有探索者电子模块，使用方便
- 所有 3P、4P 接口采用防反插设计，避免电子模块间连线造成的误操作
- 板载舵机接口、直流电机驱动芯片、MAX7219LED 驱动芯片，可直接驱动舵机、直流电机、数码管等机器人常规执行部件，无需外围电路
- 具有 5v、3.3v 及 vin 3 种电源接口，便于为各类扩展模块供电

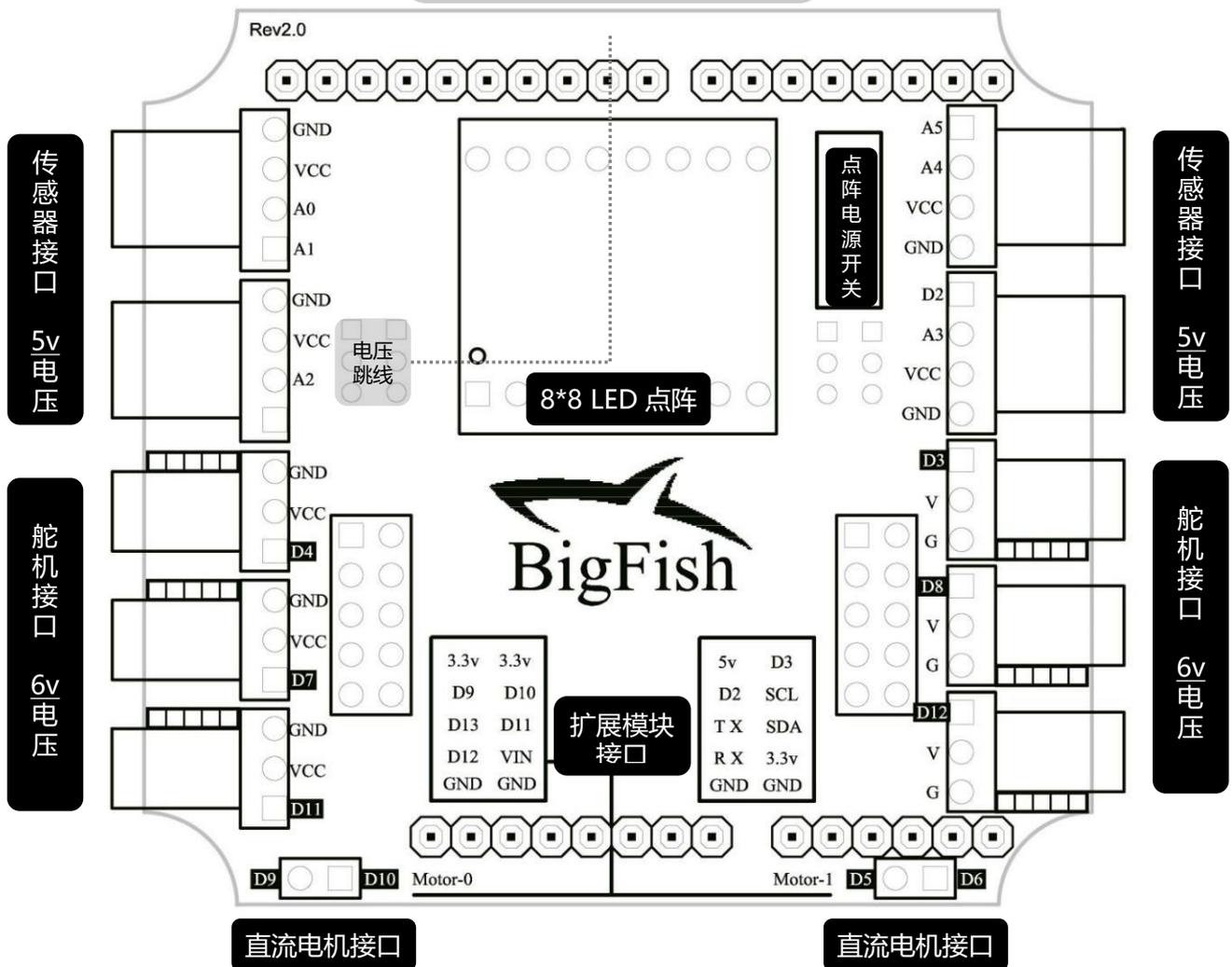
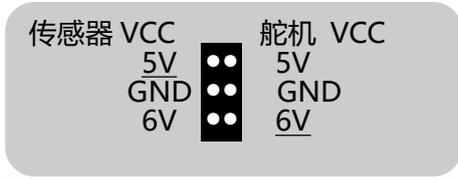
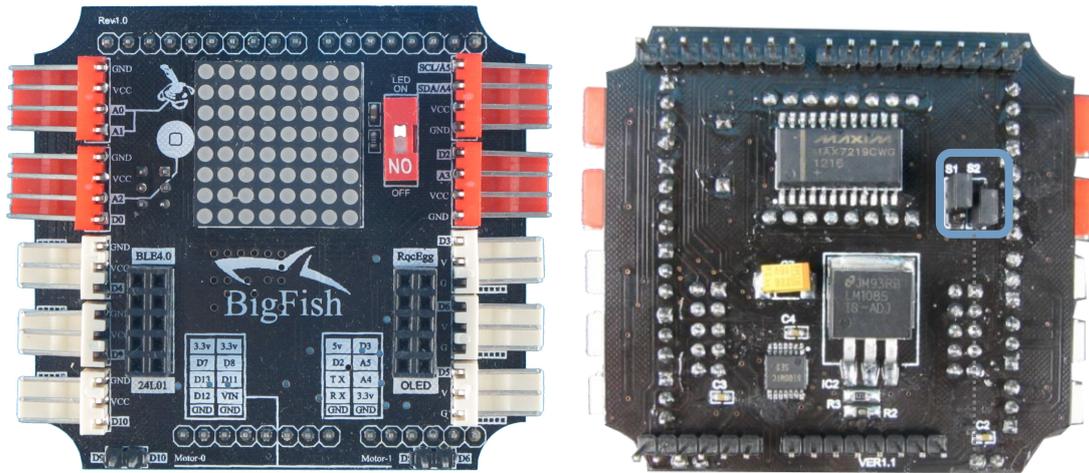
参数

- 4 针防反插接口供电 5v
- 舵机接口使用 3A 的稳压芯片 7805ADJ，为舵机提供 6v 额定电压
- 8*8led 模块采用 MAX7219 驱动芯片
- 板载一片直流电机驱动芯片 FAN8100MTC，支持 5v~9v 的 vin 电压，可同时驱动两个直流电机。
- 2 个 2*5 的杜邦座扩展坞，方便无线模块、OLED、蓝牙等扩展模块直插连接，无需额外接线

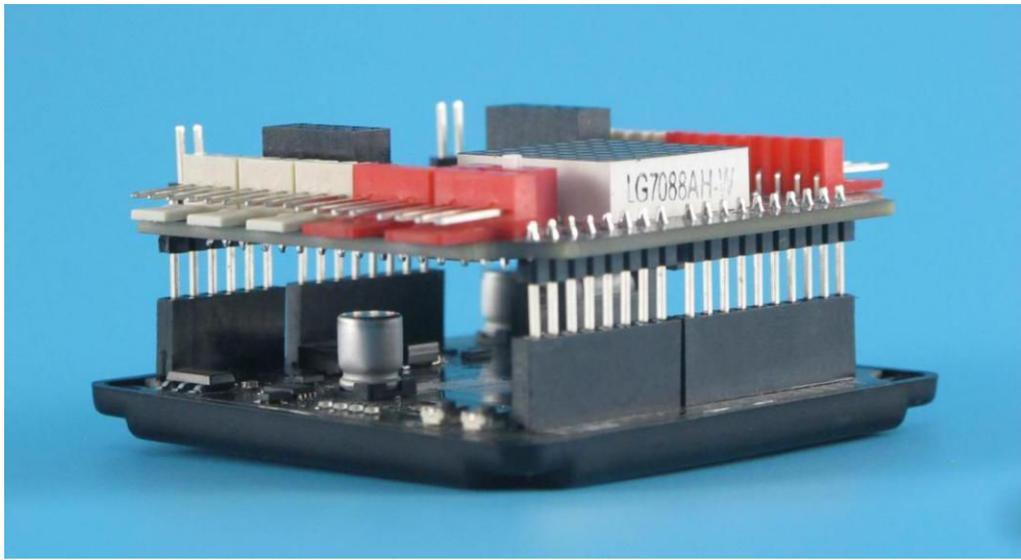
注意事项

直流电机端口与舵机端口复用，默认情况下，最多支持 6 个舵机，或 1 个直流电机 4 个舵机，或 2 个直流电机 2 个舵机。具体复用端口请见实物图片与接口。

实物图片与接口



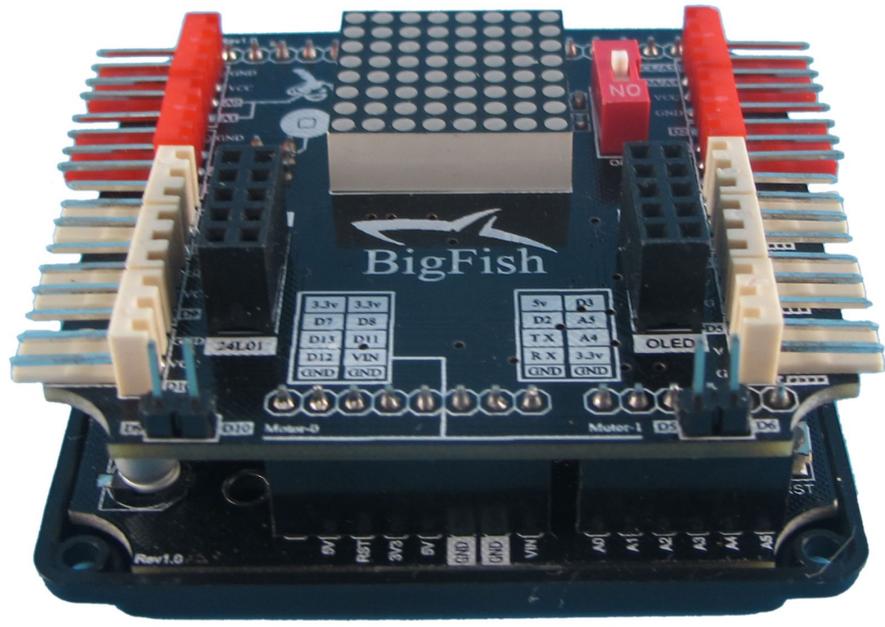
与控制板堆叠连接



BigFish 扩展板



控制板



实验 3 驱动轮模块

实验目的：1. 通过组装驱动轮模块，熟悉部分“探索者”零部件，掌握“驱动轮”的基本组装技法、组装工具使用方法，以及 STPViewer 三维浏览软件的使用方法；2. 通过搭建控制电路，掌握“探索者”基本电路的连接方法；3. 通过控制驱动轮模块，掌握 Arduino 软件的基本使用方法，尝试图形化编程，C 语言编程；4. 认识 digitalWrite 函数，analogWrite 函数；5. 熟悉驱动轮模块 module003 的运动性能。

实验性质：验证型实验

实验课时：2 课时

参考资料：请参考**光盘**中的 STP 装配文件，000_module003_dw.stp，以及光盘**课件**“基础零件组装”部分相关 PPT，尤其是“**直流电机的安装**”和“**橡胶轮胎与联轴器的安装**”，以及“软件的安装与使用”部分的“**STPViewer 三维浏览软件的使用**”。你需要用到直流电机、直流电机支架、直流电机输出头、联轴器、1:10 模型轮胎、螺丝螺母，以及组装工具等。

你还需要用到光盘中的“**探索者**”**软件套装**文件夹中的软件。请参考课件中“基础电路连接”部分的“**Bigfish 连接直流电机**”、“软件的安装与使用”部分的“**Mehran 在 Windows 环境下配置与烧录**”，连好部件，设置好软件。

光盘中的 PPT 资料给出了非常详细的基础操作说明，请大家在课下预览，上课时方便查阅，在以后的实验项目中，我们只给出简单的参考资料提示，不再详细展开。

实验步骤：

1. 组装一个驱动轮模块 (编号 : 000_module003_dw);



图 3.1 驱动轮模块 module003

结构说明：这种驱动轮模块原理非常简单，就是将 1:10 模型轮胎通过联轴器安装在 1:87 直流减速电机的输出头上，初次组装有一定难度。

运动特性：输出转矩，轮子转动的角速度、转动方向与电机一致，线速度一般，力量一般，摩擦力较好。

2. 将 Mehran 主控板、Bigfish 扩展板、锂电池和驱动轮模块连接成电路；
3. 在图形化编程界面 Ardublock 中编写以下程序并烧录；



图 3.2 驱动轮供电转动写法 1

当直流电机连在 D9/D10 针脚 (Bigfish 下方左侧的直流接口) 时 , 可以通过把 D9 或 D10 置高来供电。

此时 , 图 3.2 程序的写法和图 3.3 程序的写法是等价的。



图 3.3 驱动轮供电转动写法 2

你还会发现 , 按过 “上载到 Arduino” 按钮之后 , C 语言界面上自动生成了 C 语言代码。对应生成的 C 语言代码分别为 :

```
void setup()
{
  pinMode( 9 , OUTPUT);
}

void loop()
{
  digitalWrite( 9 , HIGH );
}
```

和

```
void setup()
{
  pinMode( 9 , OUTPUT);
  pinMode( 10 , OUTPUT);
}

void loop()
{
  digitalWrite( 9 , HIGH );
  digitalWrite( 10 , LOW );
}
```

digitalWrite 有两个参数，很容易掌握，请对应图形程序观察、学习。

4. 请分别更改供电端口号为 10、5、6，观察驱动轮模块转动的情况，以顺时针或逆时针来记录。

5. 在图形化编程界面 Ardublock 中编写以下程序并烧录；



图 3.4 驱动轮转速的控制

生成的 C 语言代码为：

```
void setup()
{
  pinMode( 9, OUTPUT);
}

void loop()
{
  analogWrite(9 , 255);
}
```

在这种写法下，可以利用 analogWrite 函数，通过改变 PWM 占空比来改变电机的转动速度。analogWrite 函数通过 PWM 的方式在引脚上输出一个模拟量，较多的应用在 LED 亮度控制、电机转速控制等方面。analogWrite 有两个参数 pin 和 value，参数 pin 表示所要设置的引脚，只能选择函数支持的引脚；参数 value 表示 PWM 输出的占空比，范围在 0~255 的区间，对应的占空比为 0%~100%。

请大家修改 value，观察模块运动的变化情况。在实际中，由于有负载，当 value 低于某值时，就已经带不动电机了，不需要 value 取 0。

实验 4 双轮万向车

实验目的：1. 通过一个电机调速例程，学习用 for 循环编写底盘调速程序；2.学会调用 Arduino 本身的例程库；

实验性质：设计型实验

实验课时：2 课时

参考资料：023_Omni-directional wheel_2m.stp

实验步骤：

1. 组装一个 023 机构；

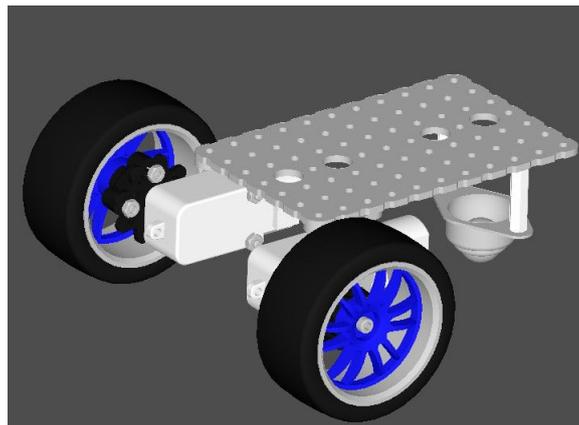


图 4.1 023 号机构——双轮万向底盘

结构说明：023 号机构底盘支点部分使用了牛眼万向轮，运动性能更好，基本是个成熟的底盘方案了。

运动特性：能完成前进、后退、左转、右转、原地旋转等动作，比较稳定，转动时采用的是差速转动。如果想更加稳定，还可以在对面再增加一个万向轮，改装成下面的结构。

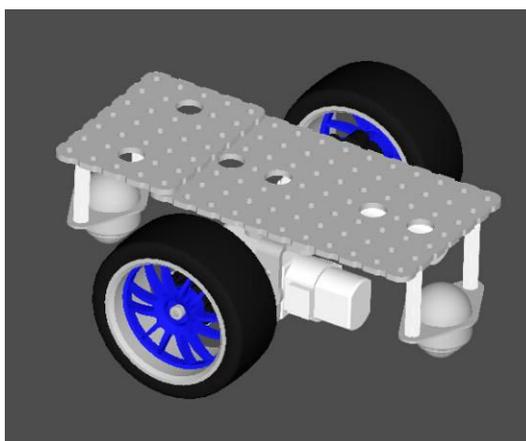


图 4.2 改装后的机构 023b

2. 按下图步骤打开例程 Fading 并烧录；

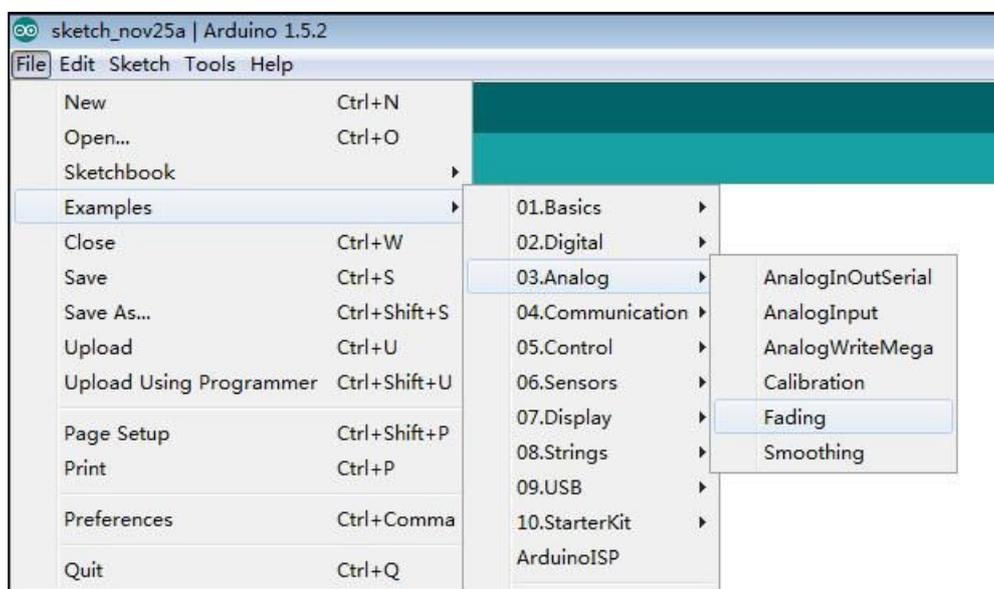


图 4.3 打开 Arduino 自带例程

主程序为：

```
int ledPin = 9;    // LED connected to digital pin 9

void setup() {
  // nothing happens in setup
}

void loop() {
  // fade in from min to max in increments of 5 points:
```

```
for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5) {  
    // sets the value (range from 0 to 255):  
    analogWrite(ledPin, fadeValue);  
    // wait for 30 milliseconds to see the dimming effect  
    delay(30);  
}  
  
// fade out from max to min in increments of 5 points:  
for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5) {  
    // sets the value (range from 0 to 255):  
    analogWrite(ledPin, fadeValue);  
    // wait for 30 milliseconds to see the dimming effect  
    delay(30);  
}  
}
```

这个例程本身是控制 LED 的，我们也可以用它来控制直流电机。烧录后观察 D9/D10 端口驱动轮转动的情况。

3. 改写这段代码，生成一段新的程序，使机器人实现忽快忽慢的周期运动。
4. 更换一部分车身零件，或改变现有零件的组装方式，使机器人结构在同样原理下达到不同的外观效果。

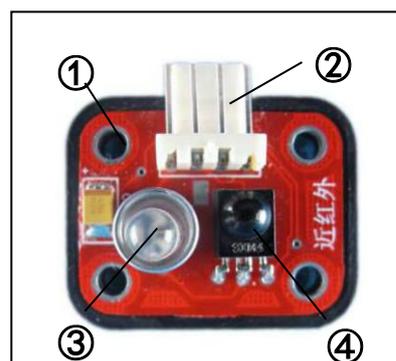
传感器简介

数字型传感器，只能返回 0 或 1，也就是高电平信号或者低电平信号，比较类似一个电源的开或关，所以也被称作开关量传感器。这类传感器都是低电平触发，也就是说，触发时产生一个低电平信号；换句话说，传感器发出低电平信号时，主控板将这个信号标为 1，高电平时为 0。“探索者”设备里有 8 种数字量传感器。

近红外传感器

近红外传感器可以发射并接收反射的红外信号，有效检测范围在 20cm 以内。工作电压：4.7~5.5V，工作电流：1.2mA，频率 38KHZ。

- ①固定孔，便于用螺丝将模块固定于机器人上
- ②四芯输入线接口，连接四芯输入线
- ③近红外信号发射头，用于发射红外信号
- ④近红外信号接收头，用于接收反射的红外信号



工作原理：发射并接收红外线，与黑标类似，但是发射功率更大，即使物体是黑色也能大量反射，因此能识别到物体，多用于避障、物体检测等。

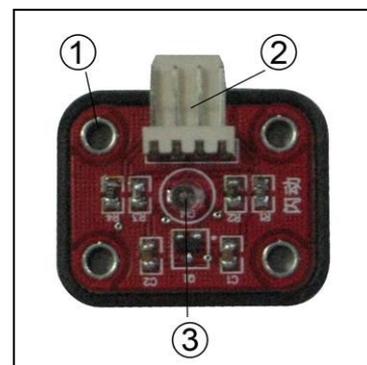
注意事项：在安装近红外传感器时，注意不要遮挡发射和接收头，以免传感器检测发生偏差。

近红外传感器触发距离较近，而且不可调整触发距离，可以调整触发距离的是另一种：“红外测距传感器”。近红外比较容易受到环境光线的干扰，比如正午的阳光，距离较近的日光灯等，都会影响其触发距离，或者对其误触发。使用时需要注意。

闪动传感器

闪动传感器可以检测到环境光线的突然变化，从而使机器人做出相应的指令动作。30 LUX 照度以上变暗触发，30LUX 照度以下变亮触发。可通过用手电筒照射或者用手遮挡光线均可触发。

- ①固定孔，便于用螺丝将模块固定于机器人上
- ②四芯输入线接口，连接四芯输入线
- ③光敏元件，检测光线强度



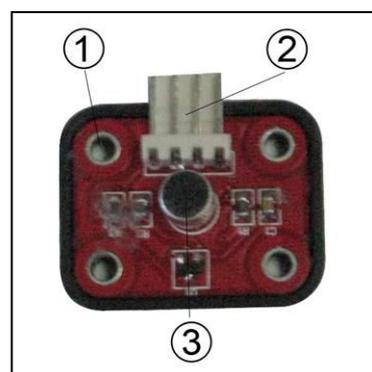
注意事项：日光灯是有闪烁的，频率在 50HZ 左右，这种闪烁会被闪动传感器识别，因此要避免在日光灯下使用。

闪动传感器是一种光敏传感器，由于光线明暗的瞬间变化，甚至日光灯的闪烁频率都能被其识别，因此非常灵敏，可以用在需要灵敏触发的场合。

声控传感器

声控传感器可以检测到周围环境的声音信号，声控元件是对震动敏感的物质，有声音时就触发。有效检测范围在 50 分贝以上（参考正常人说话时的声音）。

- ①固定孔，便于用螺丝将模块固定于机器人上
- ②四芯输入线接口，连接四芯输入线
- ③微型麦克风，检测声音

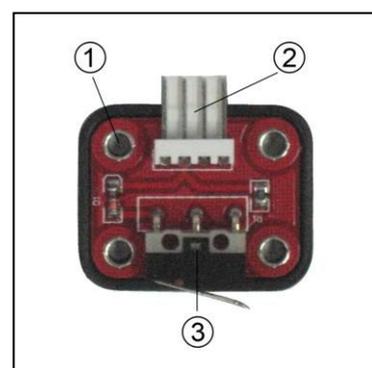


注意事项：声控传感器需要安装在较安静的机器人部位，如离电机较远的位置，最好有螺柱等与机器人本体隔离，否则特别容易被触发。

触碰传感器

触碰传感器可以检测物体对开关的有效触碰，通过触碰开关触发相应动作。触碰开关行程距离 2mm。

- ①固定孔，便于用螺丝将模块固定于机器人上
- ②四芯输入线接口，连接四芯输入线
- ③触碰开关，检测触碰

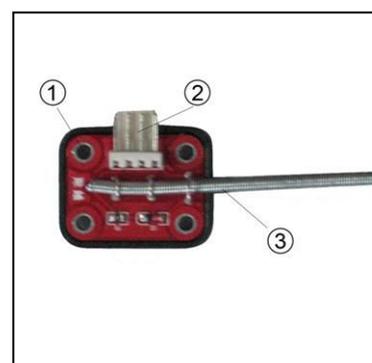


注意事项：触碰感应器需要安装在机器人容易被触碰到的位置，需要触碰开关本身被物体碰到后才会被触发。

触须传感器

触须传感器可以检测到物体对弹簧触须的有效触动。安装时通常是将弹簧与地面平行。有效触动角度 45 度。

- ①固定孔，便于用螺丝将模块固定于机器人上
- ②四芯输入线接口，连接四芯输入线
- ③弹簧触须：与障碍物接触后发生弹性形变，触发传感器



注意事项：触须感应器需要安装在机器人前端容易被触碰到的位置，需要弹簧触须被物体折弯至接触金属卡桥才会被触发。

除了上述 5 种数字量传感器，还有 3 种传感器比较特殊，即黑标、白标、光强。这三种传感器即可以作数字量传感器用，也可以作模拟量传感器用（模拟量传感器的内容将在 6.2 中讲到）。

黑标/白标传感器

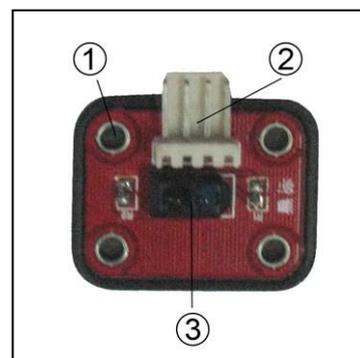
黑标/白标传感器可以帮助进行黑线/白线的跟踪，可以识别白色/黑色背景中的黑色/白色区域，或悬崖边缘。寻线信号可以提供稳定的输出信号，使寻线更准确更稳定。**有效距离在 0.7cm~3cm 之间。**

工作电压：4.7 ~ 5.5V，工作电流：1.2mA。

①固定孔，便于用螺丝将模块固定于机器人上

②四芯输入线接口，连接四芯输入线

③黑标/白标传感器元件，用于检测黑线/白线信号



工作原理：图为黑标传感器，③号元件是一个红外发射/接收管（蓝发黑收），可以发射红外线并接收反射的红外线，如果目标颜色较深，红外线就会被吸收，从而触发。因此，如果目标是开阔空间，没有红外线反射回来，也会触发。白标触发原理与黑标相反。

注意事项：黑标/白标传感器的安装应当贴近地面且与地面平行，使用前最好测试一下触发距离，这样才能更加灵敏并且有效的检测到信号。

有用户可能了解过，能够用于循迹的传感器不止一种，比如有种叫做“光电传感器”的，能发出红光，可经常用于循迹。所谓“光电传感器”是指发出光信号，并接收发射的光信号，然后将反射的光信号转化为电信号。说到这里我们就能明白，其实黑标和白标也是一种光电传感器，是红外光电，红外线不可见；而那种能看到红光的则是使用可见光的光电传感器。

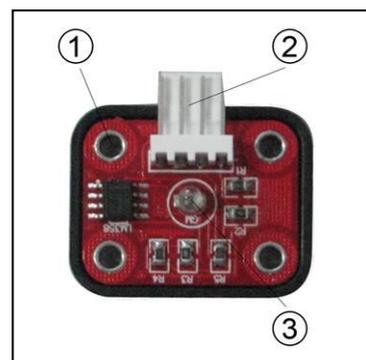
由于生产厂家众多再加上用户群体不同，同一种传感器可能会出现不同的命名，用户在使用时要注意了解传感器的工作原理，从而掌握更多的传感器。

光强传感器

光强传感器可以检测到周围光线强度的变化。光强传感器能够识别光线强弱，闪动传感器只能检测光线的突变。30 LUX 照度以下触发。

(距离 40 瓦日光灯 1.5 米左右)

- ①固定孔，便于用螺丝将模块固定于机器人上
- ②四芯输入线接口，连接四芯输入线
- ③光敏元件，当光线由强变弱时被触发



注意事项：安装时注意将感光元件对准光源。这样传感器才能较精确的检测到光线的强弱变化。

黑标/白标传感器

当作为模拟量传感器使用时也叫灰度传感器，它们可以检测到不同的灰阶。从而不仅仅可以识别黑色白色，还可以识别灰度属性不同的其他颜色。在某些机器人比赛中，会用由黑到白的色彩布置场地，供机器人识别，这时候就是灰度传感器大显身手的时候了。



工作原理：灰阶简单说就是把所有色彩都转化成黑与白的组合时，它们所表现出的不同深浅的灰色。结合我们了解过的这两种传感器的工作原理，我们可以明白，不同的颜色反射的红外线不同，这个数据可以转化成灰度数值。

光强传感器

光强传感器作为模拟量传感器使用时，可以检测到照度数据，即对应光线明暗程度的数据。光强传感器可以用于与光亮渐变相关的项目里，比如随天色自动变量变暗的路灯、小夜灯等。



注意事项：安装时注意将感光元件对准光源。这样传感器才能较精确的检测到光线的强弱变化。

超声测距传感器

HC-SR04 超声波测距模块可提供 2cm-400cm 的非接触式距离感测功能，测距精度可达高到 3mm；模块包括超声波发射器、接收器与控制电路。



应用范围：能够探测出距离，因此可以用于如机器人避障，距离测量、高度测量、物体表面扫描等项目。

温湿度传感器

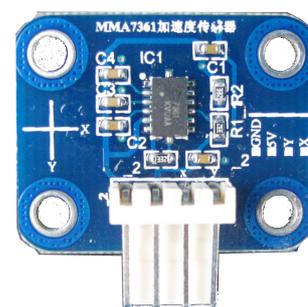
DHT11 数字温湿度传感器是一款含有已校准数字信号输出的温湿度复合传感器。传感器包括一个电阻式感湿元件和一个 NTC 测温元件。校准系数以程序的形式存在 OTP 内存中，传感器内部在检测型号的处理过程中要调用这些校准系数。单线制串行接口，使系统集成变得简易快捷。



注意事项：探测头必须与被测物体接触才能测量到温度数据。

加速度传感器

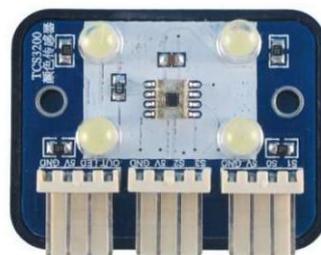
加速度传感器是一种可以对物体运动过程中的加速度进行测量的电子设备，典型互动应用中的加速度传感器可以用来对物体的姿态或者运动方向进行检测，比如 Wii 游戏机和 iPhone 手机中的经典应用。



应用范围：加速度传感器采用 Freescale (飞思卡尔) 公司生产的高性价比微型电容式三轴加速度传感器 MMA7361 芯片，但是取消了 Z 轴，所以只有两个轴。可以应用到摩托车和汽车放倒报警、遥控航模、游戏手柄、人形机器人跌倒检测、硬盘冲击保护、倾斜度测量等场合。

颜色识别传感器

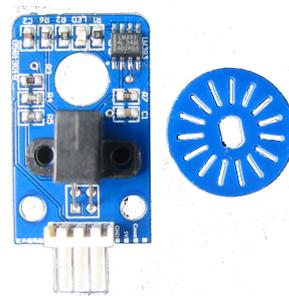
TCS3200 颜色传感器是一款全彩的颜色检测器，包括了一块 TAOS TCS3200RGB 感应芯片和 4 个白色 LED 灯，TCS3200 能在一定的范围内检测和测量几乎所有的可见光。



工作原理：TCS3200 有大量的光检测器，每个都有红绿蓝和清除 4 种滤光器。每 6 种颜色滤光器均匀地按数组分布来清除颜色中偏移位置的颜色分量。内置的振荡器能输出方波，其频率与所选择的光的强度成比例关系。

红外编码器

本模块为直流电机测速模块。采用槽型红外线对射，只要在不透明物体通过槽型即可触发 (配合码盘使用) 输出 5V TTL 电平。



工作原理：采用施密特触发器去产生脉冲，可用于计数，从而计算直流电机的转动速度，辅助完成 PID 算法等。

实验 5 触碰触须启动小车

实验目的：1. 使用一个触碰传感器实现开关功能，熟悉“探索者” Basra 主控板、Bigfish 扩展板，掌握触碰传感器的特性和使用方法；2. 通过搭建检测电路，掌握“探索者”基本检测电路的连接方法；3. 通过编写触碰开关程序，掌握 Arduino 软件的基本使用方法，尝试图形化编程，C 语言编程；4. 了解 pinMode、digitalRead、digitalWrite、delay 等函数的功能；5. 学会使用“如果/否则”逻辑。

实验性质：验证型实验

实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Mehran 控制板、BigFish 扩展板，以及**光盘课件**“基础电路连接”部分相关 PPT，如“**Basra 基础连接**”和“**Bigfish 连接常规传感器**”，以及“软件的安装与使用”部分的“**Mehran 在 Windows 环境下配置与烧录**”。

实验步骤：

1. 你需要挑选一个机器人作为执行机构。比如 023_Omni-directional wheel_2m，(以下简称“023 号机构”)

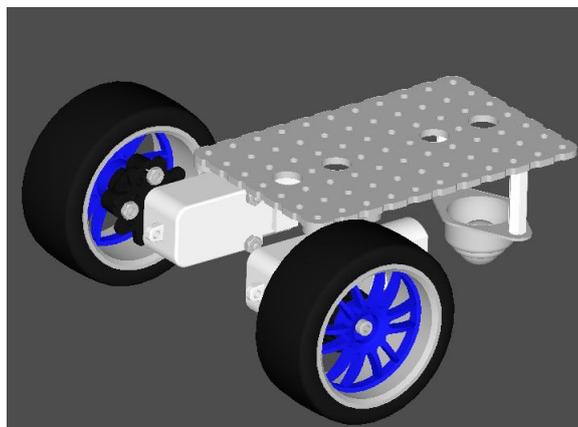


图 1.1 023_Omni-directional wheel_2m

这是一个适用性比较广泛的机构，基本所有需要底盘搭载的实验都可以用这个机构去执

行。所以这个实验，包括后面的实验，都将优先选择这个机构。当然，您也可以选择其他任何您觉得合适的机构。

如果想要更简单一些，也可以不用完整的机器人，一个小模块，甚至一个电机，一个LED也可以。如：

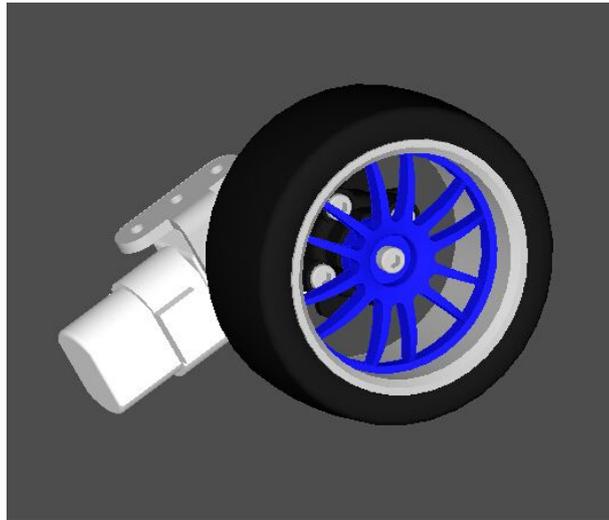


图 1.2 驱动轮模块 module003

或者



图 1.3 关节模块 module002

2. 取一个触碰传感器，安装在机构上，将 Basra 主控板、Bigfish 扩展板、触碰传感器、电机等连好电路。

3. 在图形化编程界面 Ardublock 中编写以下程序并烧录：

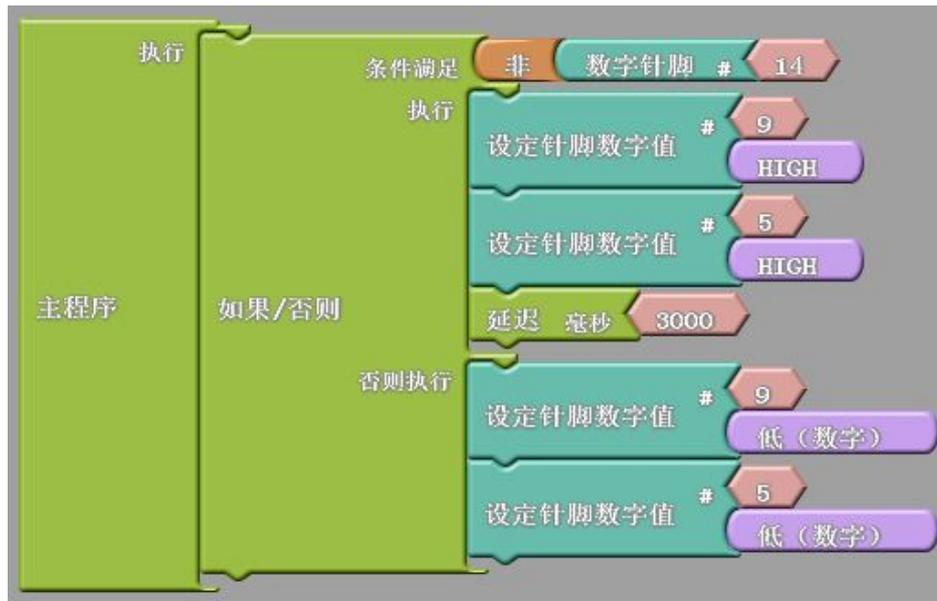


图 1.4 触碰开关启动双轮小车

TTL 传感器的触发条件是“低电平”，而  语句意思是“14 号数字引脚获得高电平”，因此要在前面加一个逻辑运算符 

生成的 C 语言代码为：

```

void setup()
{
  pinMode( 14, INPUT);
  pinMode( 9 , OUTPUT);
  pinMode( 5 , OUTPUT);
}

void loop()
{
  if (!( digitalRead(14) ))
  {
    digitalWrite( 9 , HIGH );
    digitalWrite( 5 , HIGH );
    delay( 3000 );
  }
  else
  {
    digitalWrite( 9 , LOW );
    digitalWrite( 5 , LOW );
  }
}
  
```

改变延时参数，重新烧录，观察对运行效果的影响，尤其是当延时参数小到一定程度后的运行效果。然后去掉延时语句，对比观察，并思考为什么。

实验 6 红外启动小车

实验目的：1. 使用一个其他类别 TTL 传感器实现开关功能，掌握近红外、光强、闪动、声控、火焰等传感器的特性和使用方法；2. 通过搭建检测电路，掌握“探索者”基本检测电路的连接方法；3. 了解近红外传感器安装时需要注意的事项。4. 了解传感器的使用环境以及干扰因素。5. 进一步熟练图形化编程及 C 语言编程的方法。

实验性质：设计型实验

实验课时：2 课时

参考资料：请参考光盘中的“电子元件资料”，Basra 控制板、BigFish 扩展板，以及光盘课件“基础电路连接”部分相关 PPT，如“basra 基础连接”和“Bigfish 连接常规传感器”，以及“软件的安装与使用”部分的“basra 在 Windows 环境下配置与烧录”。

实验步骤：

1. 挑选一个机构，分别为其安装近红外、光强、闪动、声控等开关，注意观察传感器的触发条件，保证安装后可以正常使用。
2. 将实验 1 的图形程序或代码移植过来使用。
3. 分别在无光照条件下，正午阳光条件下，日光灯条件下使用近红外传感器，观察传感器的工作效果。
4. 用手机摄像头观察近红外传感器在工作状态和非工作状态的区别（根据手机不同显示效果不同）。
5. 对代码进行改写，反转光强传感器的触发条件。
6. 在日光灯条件下使用闪动传感器，观察传感器的工作效果。
7. 用声控传感器实现声音开关功能。
用火焰传感器实现火焰开关功能。

实验 7 带传动模块

实验目的：1. 熟悉可拆装式履带的结构特点和组装规律；2. 学会组装带轮和履带。

实验性质：验证型实验

实验课时：2 课时

参考资料：000_module005_cb.stp；026a_Tracked vehicle_2m.stp；026b_Tracked vehicle_2m；履带的组装与拆卸.ppt

实验步骤：

1. 组装履带模块

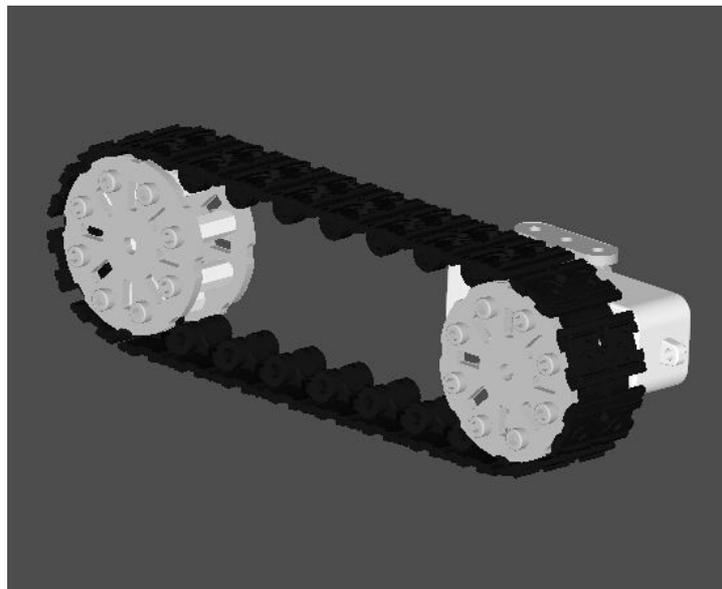


图 7.1 履带模块 module005

结构说明：履带可以传送转矩到随动轮，从而让前后两组轮子都能驱动。运动能力更强，爬坡、翻越障碍的能力更强。可拆卸的履带片方便使用者调整设计方案，从而构造不同长度和造型的履带。图中履带更像是传送带，对于模型底盘、小型工程机械已经够用了，但是在实际工程中，往往需要增加更多的带轮、负重轮或者张紧轮，或者更换履带材质。

运动特性：转动灵活，摩擦力好，地形适应能力强。

2. 将履带模块安装于一个自制的架子上，通过增加或减少履带片，改变履带的长度，或调

节履带的松紧度。记录松紧度最佳时的履带片数量和带轮的圆心孔距。

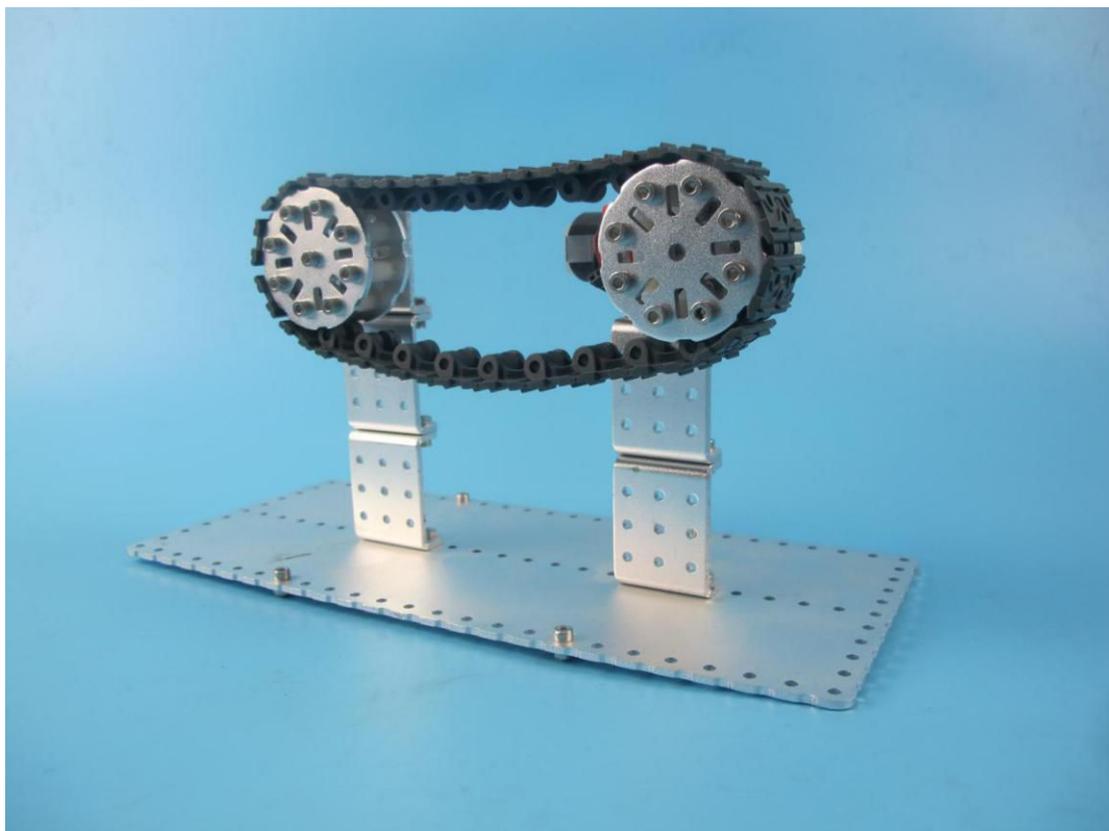


图 7.2 带传动机构

实验 8 履带机器人

实验目的：1. 熟悉履带模块在底盘机构中的应用。

实验性质：验证型实验

实验课时：2 课时

参考资料：000_module005_cb.stp；026a_Tracked vehicle_2m.stp；026b_Tracked vehicle_2m；履带的组装与拆卸.ppt

实验步骤：

1. 组装履带机器人

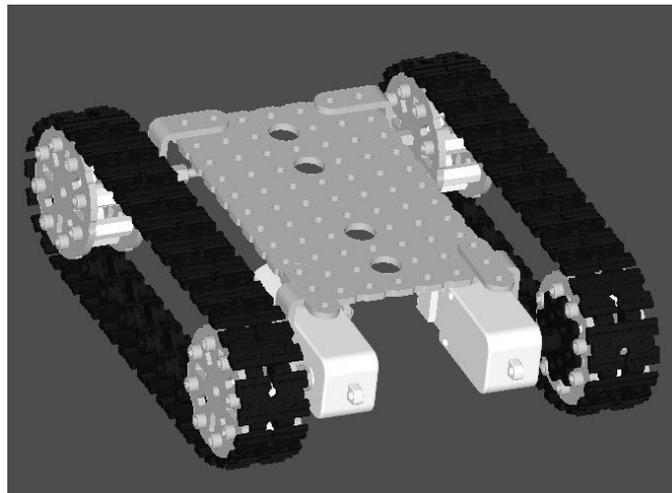


图 8.1 履带车 026a

结构说明：履带可以传送转矩到随动轮，从而让前后两组轮子都能驱动。运动能力更强，爬坡、翻越障碍的能力更强。可拆卸的履带片方便使用者调整设计方案，从而构造不同长度和造型的履带。图中履带更像是传送带，对于模型底盘、小型工程机械已经够用了，但是在实际工程中，往往需要增加更多的带轮、负重轮或者张紧轮，或者更换履带材质。

运动特性：转动灵活，摩擦力好，地形适应能力强。

2. 尝试控制它运动，控制方式和控制一个双轮车是一样的。
3. 调整履带长度，组装一个迷你履带车，并尝试控制它运动

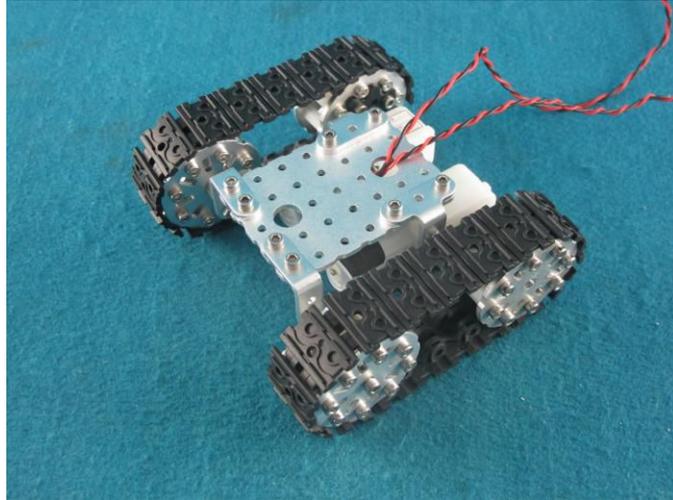


图 8.2 迷你履带车 026b

实验 9 红绿灯

实验目的：1. 了解一种信号灯的实现方式；2. 了解 LED 模块的工作特性及安装注意事项。

3. 了解四芯输出线的连接方法。

实验性质：验证型实验

实验课时：2 课时

参考资料：请参考 Bigfish 连接常规传感器.ppt；

主要器材：LED 模块×1

实验步骤：

1. 将 LED 模块连上四芯输出线，然后连接在 Bigfish 的 A4/A5 传感器端口上，注意 GND；

2. 将以下程序烧录进主控板，它将实现以下效果：红灯、绿灯以 1 秒为间隔，交替闪烁。



生成的 C 语言为：

```
void red();
void green();
void off();

void setup()
{
  pinMode( 18 , OUTPUT);
  pinMode( 19 , OUTPUT);
}

void loop()
{
  red();
  delay( 1000 );
  off();
  delay( 1000 );
  green();
  delay( 1000 );
  off();
  delay( 1000 );
}

void red()
{
  digitalWrite( 18 , HIGH );
  digitalWrite( 19 , LOW );
}

void off()
{
  digitalWrite( 18 , LOW );
  digitalWrite( 19 , LOW );
}

void green()
{
  digitalWrite( 18 , LOW );
  digitalWrite( 19 , HIGH );
}
```

3. 尝试用其他端口控制 LED 模块；
4. 参考实验 3 第 5 点，改写程序，尝试控制 LED 灯亮度强弱。

实验 10 声控灯

实验目的：1. 使用一个声控传感器实现开关功能，熟悉“探索者”主控板、扩展板，掌握舵机的特性和使用方法；2. 学会寻找舵机中值；3. 通过编写声控开关程序，掌握 Arduino 软件的基本使用方法，尝试图形化编程，C 语言编程；4. 了解 servo 类函数的功能和使用方法；5. 学会使用“如果/否则”逻辑。

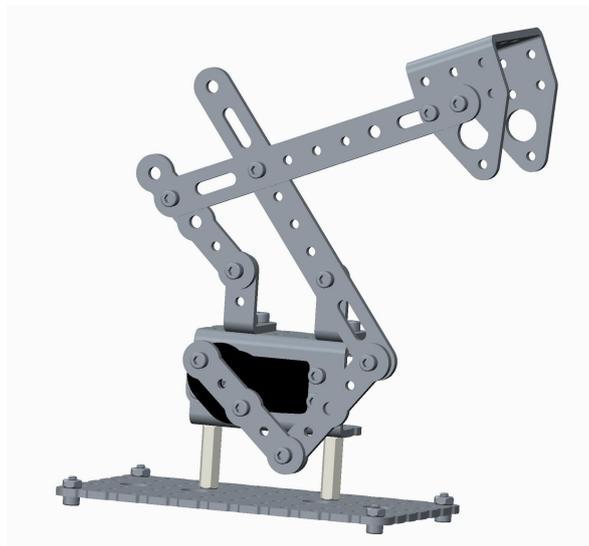
实验性质：验证型实验、设计型实验

实验课时：2 课时

参考资料：请参考光盘中的“电子元件资料”，Basra 控制板、BigFish 扩展板，以及光盘课件“基础电路连接”部分相关 PPT，如“[basra 基础连接](#)”和“[Bigfish 连接常规传感器](#)”，以及“软件的安装与使用”部分的“[basra 在 Windows 环境下配置与烧录](#)”。

实验步骤：

1. 你需要挑选一个机器人作为执行机构。



这是一个舵机（小型伺服电机）驱动的机构，它的运动特点是只能左右摆动，运动范围为 $\pm 90^\circ$ ，伸缩功能由连杆组完成。由于有范围限制，那么安装输出头的时候就不能像直流电机那样随便安装了，而必须找到合适的角度位置安装。

2. 取一个声控传感器，安装在机构上，将 basra 主控板、Bigfish 扩展板、声控传感器、电机等连好电路。

3. 在图形化编程界面 Ardublock 中编写以下程序并烧录：



生成的 C 语言代码为：

```
void setup()
#include <Servo.h>

Servo servo_pin_3;

void setup()
{
  pinMode( 14, INPUT);
  servo_pin_3.attach(3);
  pinMode( 18 , OUTPUT);
  pinMode( 19 , OUTPUT);
}

void loop()
{
  if (!( digitalRead(14) ))
  {
    servo_pin_3.write( 60 );
    digitalWrite( 18 , HIGH );
```

```
        digitalWrite( 19 , LOW );  
        delay( 1000 );  
    }  
    else  
    {  
        servo_pin_3.write( 90 );  
        digitalWrite( 18 , LOW );  
        digitalWrite( 19 , LOW );  
    }  
}
```

4. 改写程序，使机构展开后灯再亮，以及灯熄灭后，机构再收拢。

实验 11 光感小夜灯

实验目的：1. 使用一个光强传感器实现光控开关功能，熟悉“探索者”Mehran 主控板、Bigfish 扩展板，掌握光强传感器的特性和使用方法；2. 通过搭建检测电路，掌握“探索者”基本检测电路的连接方法；3. 通过编写光控开关程序，掌握 Arduino 软件的基本使用方法，尝试图形化编程，C 语言编程；4. 了解 pinMode、digitalRead、digitalWrite、delay 等函数的功能；5. 学会使用“如果/否则”逻辑。

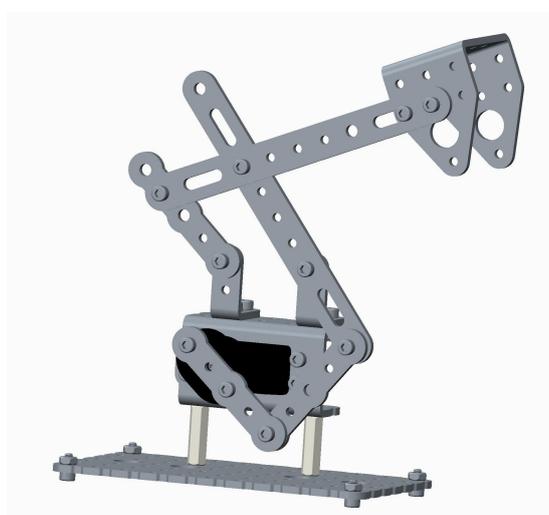
实验性质：验证型实验

实验课时：2 课时

参考资料：请参考光盘中的“电子元件资料”，Basra 控制板、BigFish 扩展板，以及光盘课件“基础电路连接”部分相关 PPT，如“Basra 基础连接”和“Bigfish 连接常规传感器”，以及“软件的安装与使用”部分的“Basra 在 Windows 环境下配置与烧录”。

实验步骤：

1. 挑选伸缩台灯作为执行机构。它将达到这样的效果：当环境光线比较明亮时，台灯处于“收拢并熄灭”的状态，当环境光线比较暗时，台灯伸展并点亮。你可以用物体遮挡传感器使环境光线发生变化。如果周围环境比较暗，你可以用手机上的手电筒使环境光线发生变化。



2. 取一个光强传感器，安装在机构上，将 Basra 主控板、Bigfish 扩展板、触碰传感器、电机等连好电路。

3. 在图形化编程界面 Ardublock 中编写以下程序并烧录：



生成的 C 语言代码为：

```
void setup()
#include <Servo.h>

Servo servo_pin_3;

void setup()
{
  pinMode( 14, INPUT);
  servo_pin_3.attach(3);
  pinMode( 18 , OUTPUT);
  pinMode( 19 , OUTPUT);
}

void loop()
{
  if (!( digitalRead(14) ))
  {
    servo_pin_3.write( 60 );
    digitalWrite( 18 , HIGH );
```

```
        digitalWrite( 19 , LOW );  
        delay( 1000 );  
    }  
    else  
    {  
        servo_pin_3.write( 90 );  
        digitalWrite( 18 , LOW );  
        digitalWrite( 19 , LOW );  
    }  
}
```

4. 改写代码，使 LED 灯缓慢点亮，缓慢熄灭。

实验 12 红外迎宾装置

实验目的：1. 使用一个近红外传感器实现开关功能，熟悉“探索者”主控板、扩展板，掌握舵机的特性和使用方法；2. 学会寻找舵机中值；3. 通过编写近红外开关程序，掌握 Arduino 软件的基本使用方法，尝试图形化编程，C 语言编程；4. 了解 servo 类函数的功能和使用方法；5. 学会使用“如果/否则”逻辑。

实验性质：验证型实验

实验课时：2 课时

参考资料：请参考光盘中的“电子元件资料”，Basra 控制板、BigFish 扩展板，以及光盘课件“基础电路连接”部分相关 PPT，如“Basra 基础连接”和“Bigfish 连接常规传感器”，以及“软件的安装与使用”部分的“Basra 在 Windows 环境下配置与烧录”。

实验步骤：

1. 你需要挑选一个机器人作为执行机构。



这是一个侧重于外观的机构，由一个舵机驱动，结构并不复杂。当然，您也可以选择其

他任何您觉得合适的机构。

2. 取一个近红外传感器，安装在机构上，将主控板、扩展板、传感器、电机等连好电路。

3. 在图形化编程界面 Ardublock 中编写以下程序并烧录：



生成的 C 语言代码为：

```
#include <Servo.h>

Servo servo_pin_3;

void setup()
{
  pinMode( 14, INPUT);
  servo_pin_3.attach(3);
}

void loop()
{
  if (!( digitalRead(14) ))
  {
    servo_pin_3.write( 140 );
    delay( 2000 );
  }
  else
  {
    servo_pin_3.write( 90 );
  }
}
```

4. 改写代码，让鞠躬的动作更加柔和。

实验 13 机器人简单循迹

实验目的：1. 使用黑标/灰度传感器实现简单循迹功能，熟悉循迹功能的要点和注意事项；
2. 学会铺设循迹场地；3. 学会使用“如果”逻辑设置循迹程序。

实验性质：验证型实验

实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板、BigFish 扩展板，以及光盘**课件**“基础电路连接”部分相关 PPT，如“Basra 基础连接”和“Bigfish 连接常规传感器”，以及“软件的安装与使用”部分的“Basra 在 Windows 环境下配置与烧录”。

实验步骤：

1. 将两个黑标传感器安装在 023 号机构的底部前端，传感器距离车轮越远效果越好，具体位置请自己尝试，连好电路；
2. 在白色场地上用黑色绝缘胶带（在两个灰度传感器之间，越宽效果越好）铺设一条轨迹，直线、弧线、圆均可；

这种循迹方案的运动原理：

要想识别地面上的黑线或者白线，很容易可以想到使用黑标传感器，而且至少要有 2 个黑标传感器。当只安装一个传感器时，一旦小车偏离轨迹就不好办了，所以还要想办法在小车快要离开轨迹的时候把它拉回来，这样就需要另外一个传感器。

所以我们最少要用到两个黑标传感器，一个安装在车头左侧，一个安装在车头右侧，如果左侧传感器检测到轨迹，就向右行驶来纠正；同理，如果右侧传感器检测到轨迹，就向左行驶来纠正。这样就保证轨迹始终在两个传感器之间。

如下图所示：

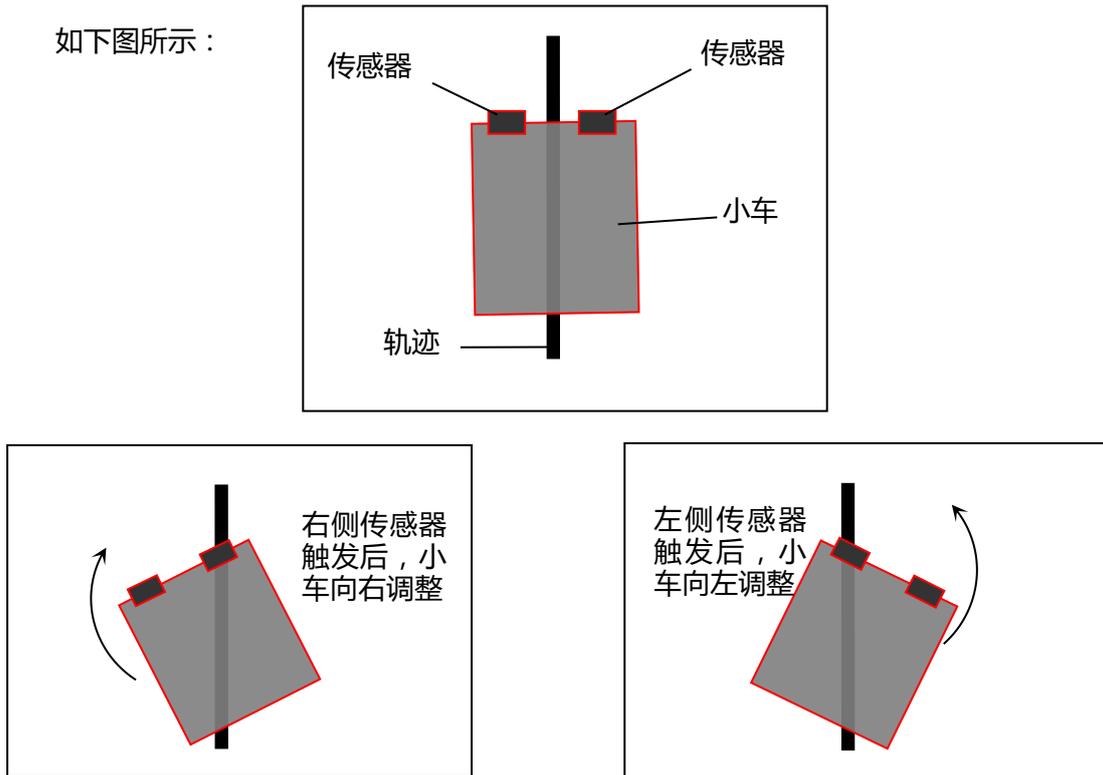


图 13.1 检测原理及程序设计设想

3. 阅读、编写并烧录下面的程序，它将实现这样一个功能：把小车放置在黑线上，小车能够自动沿着黑线行驶。



图 9.2 沿黑线行驶程序

生成 C 语言代码为：

```
void setup()
{
  pinMode( 18, INPUT);
  pinMode( 14, INPUT);
  pinMode( 5 , OUTPUT);
  pinMode( 9 , OUTPUT);
}
```

```

void loop()
{
  if (!( digitalRead(14) ))
  {
    digitalWrite( 5 , HIGH );
  }
  if (!( digitalRead(18) ))
  {
    digitalWrite( 9 , HIGH );
  }
}

```

4. 在白色场地上用黑色绝缘胶带铺设一条轨迹，**先尝试直线**，熟练后可尝试弧线、圆、矩形等；

5. 重要提示：

小车结构、场地、电机速度等因素，对循迹效果影响非常明显。以下要点请大家注意。

(1) **传感器与小车的车轮，距离要尽量远**，否则会造成小车转弯角度过大，可以用杆件等加长距离或改变电机的安装位置；

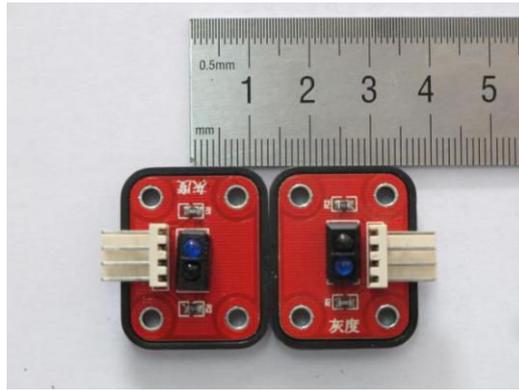
(2) 如果地面不是很平整，尽量不要把万向轮装在前进方向上，容易与地面干涉；

(3) 传感器与地面距离要在 1~3 厘米以内，简单来说，023 机构的主平板和传感器之间用 30 的螺柱就差不多了。烧录以下程序，在连接 USB 的状态下，打开 Tools→Serial Monitor，监测传感器是否可以正确触发，帮助确定正确的传感器安装位置以及场地。



这个程序可以调用串口显示功能，在 Serial Monitor 中显示 A0 端口传感器触发状态，未触发显示 0，触发显示 1。

(4) 黑标场地背景颜色要尽量浅，最好就是白色，要尽量平整。对于 2 个传感器的小车来说，黑线的宽度很重要（2 个以上传感器，黑线的宽度就无所谓了）。由于两个“探索者”黑标传感器的检测头最近也要有 2.5cm 左右的距离，因此黑线不应小于 2.5cm，否则会造成小车转弯角度太大。



(5) 两个传感器安装时尽量靠近，传感器检测头的距离应和黑线宽度相当。

(6) 循迹是一种非常综合的机器人实验。对综合调试能力要求很高。就这个实验来说，程序很简单，但是小车要走好却很难。小车的结构、传感器安装、传感器触发条件、场地状况的综合调试，比程序本身重要。大家在这个实验中要把这些程序外因素的调试训练到位。

6. 阅读、编写并烧录下面的程序，它将实现这样一个功能：把小车放置在黑背景下的白线上，小车能够自动沿着白线行驶。



图 9.3 沿白线行驶程序

生成的 C 语言代码为：

```

void setup()
{
  pinMode( 18, INPUT);
  pinMode( 14, INPUT);
  pinMode( 5 , OUTPUT);
  pinMode( 9 , OUTPUT);
}

void loop()
{
  if (digitalRead(14))

```

```
{  
  digitalWrite( 5 , HIGH );  
}  
if (digitalRead(17))  
{  
  digitalWrite( 9 , HIGH );  
}  
}
```

实验 14 机器人躲避悬崖

实验目的：使用黑标/灰度传感器实现躲避悬崖功能，理解识别悬崖功能的原理；

实验性质：设计型实验

实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板、BigFish 扩展板，以及光盘**课**
件“基础电路连接”部分相关 PPT，如“Basra 基础连接”和“Bigfish 连接常规传感器”，
以及“软件的安装与使用”部分的“Basra 在 Windows 环境下配置与烧录”。

实验步骤：

黑标传感器还有一个功能，就是可以识别悬崖，因为面向开阔空间时，没有红外线回来，
所以也会触发。所以我们可以编程实现小车遇到悬崖就后退的功能。

请大家编程实现小车行驶中遇到悬崖后退的功能。你可以使用 1 个或多个黑标传感器
实现。

实验 15 机器人悬崖漫步

实验目的：练习使用多个不同种类传感器配合实现某个功能。

实验性质：设计型实验

实验课时：2 课时

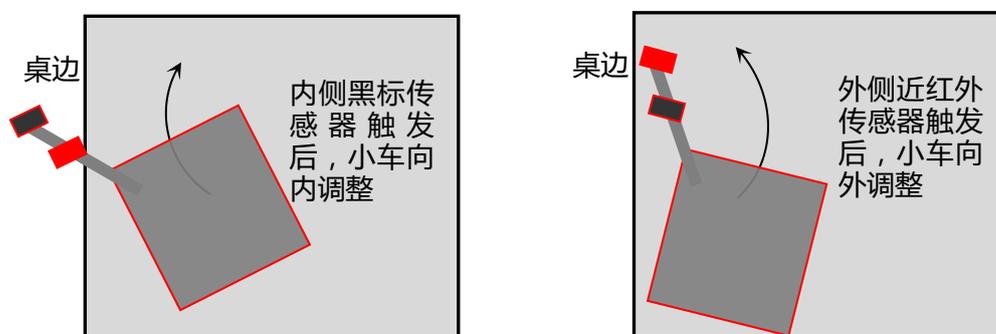
参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板、BigFish 扩展板，以及光盘**课**
件“基础电路连接”部分相关 PPT，如“**Basra 基础连接**”和“**Bigfish 连接常规传感器**”，
以及“软件的安装与使用”部分的“**Basra 在 Windows 环境下配置与烧录**”。

实验步骤：

如果要实现沿着悬崖行走的功能，那么单单使用一种传感器，实现难度可能会比较大，
如果用两个传感器配合，就会很容易实现。



沿悬崖行走的原理如下图所示：



请大家根据上图示意图所表示的原理，编程实现小车沿悬崖行走功能。

提高篇

实验 16 关节模块

实验目的：1. 熟悉机器人关节模块的结构特点和组装规律；2. 学会利用小型标准伺服电机组装关节模块；3. 熟悉关节模块在机器人结构设计中的应用；4. 熟悉标准舵机对应的函数，学会控制标准伺服电机。

实验性质：验证型实验

实验课时：1 课时

参考资料：000_module002_joint.stp；标准舵机的安装.ppt；Bigfish 连接伺服电机.ppt；

实验步骤：

1. 组装一个关节模块



图 16.1 关节模块 module002

结构说明：这种关节模块在机器人的结构设计中非常常用，思路也非常简单，就是将多个关节模块串联累加，构建成多自由度的机器人，每一个关节为一个自由度，从而实现机械臂、人形、多足仿生等机器人结构。

运动特性：能较好地模仿生物的运动形态，只要自由度足够多，几乎什么动作都能做到，常见于人形机器人舞蹈表演，或者学习机器人运动学规划等。但是结构缺少优化，缺少传动，能耗较大，做简单运动时冗余的自由度较多。在小尺度上用途非常广泛，常见于玩具、模型、

教具等，但是无法承担一定级别以上的尺寸和重量，所以工程中和工业中用途较少。

2. 将 Mehran 主控板、Bigfish 扩展板、锂电池和关节模块连接成电路；在图形化编程界面 Ardublock 中编写以下程序并烧录；



图 16.2 关节模块摆动程序的写法

这个程序意味着连在 4 号端口的舵机转到 150°的位置并始终保持。

C 语言版本为：

```
#include <Servo.h>

Servo servo_pin_4;

void setup()
{
    servo_pin_4.attach(4);
}

void loop()
{
    servo_pin_4.write( 150 );
}
```

定义针脚的函数为 `servo_pin_4.attach(4);`；

定义角度位置的参数为 `servo_pin_4.write(150);`；

3. 如果我们想让舵机来回摆动，那么就要再增加一个位置。于是把程序改成如下图所示：



图 16.3 关节模块摆动程序的写法

意思是先摆动到 150°的位置，保持 1000ms，再摆动到 30°的位置，保持 1000ms，然后不断循环。

C 语言代码为：

```

#include <Servo.h>

Servo servo_pin_4;

void setup()
{
    servo_pin_4.attach(4);
}

void loop()
{
    servo_pin_4.write( 30 );
    delay( 1000 );
    servo_pin_4.write( 150 );
    delay( 1000 );
}

```

4. 请大家把延时语句去掉，或者把延时缩短，观察运行的效果，并思考为什么会是这样。



图 16.4 去掉延时后的语句

大家会观察到，舵机并没有运行到指定的位置，而是在某个小角度范围内来回抖动。这是因为程序执行的速度很快，但是电机的转速是有极限的，电机还没来得及转到 150° ，就接到了转到 30° 的指令，必须给它足够的延时，它才能转到指定位置。

实验 17 2 自由度云台

实验目的：1. 熟悉 2 自由度云台的结构特点和组装规律；2. 学会利用关节模块组装串联结构；3. 熟悉标准舵机对应的函数，学会用 for 循环控制标准伺服电机缓慢运动，从而规划云台的扫描路径。

实验性质：验证型实验

实验课时：2 课时

参考资料：207_pan-and-tilt_2s.stp；实验 16；

实验步骤：

1. 组装一个 2 自由度云台

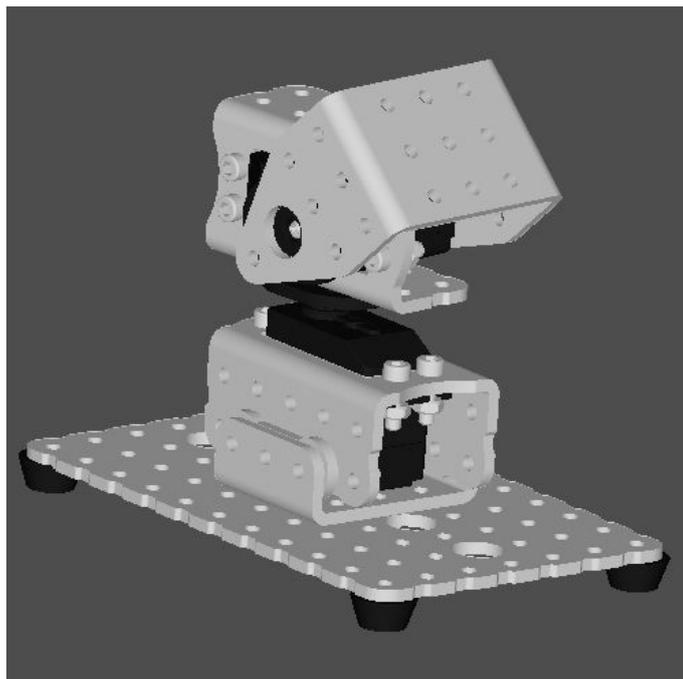
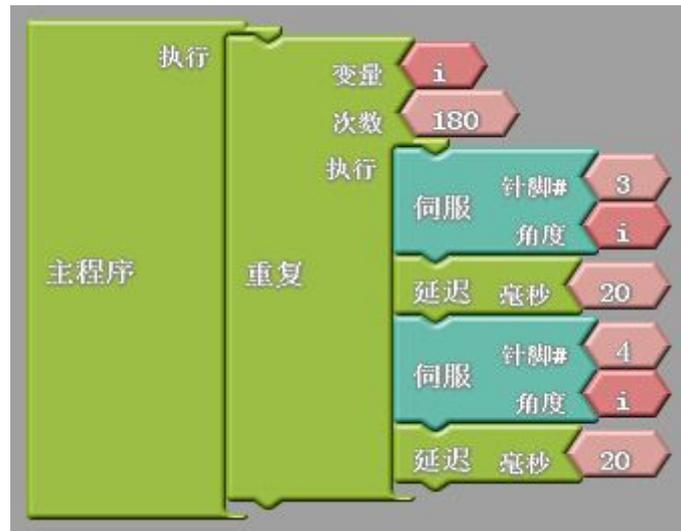


图 17.1 027 号机构——2 自由度云台

结构说明：027 号机构——2 自由度云台由 1 个标准伺服电机加一个关节模块组成，云台一般是用于搭载摄像机的结构，多用于视频监控、航拍等。

运动特性：由于关节模块行程的限制，该云台能在大约三分之一球面的范围内进行扫描，如果想要更大的扫描范围，可以对结构进行改造。

2. 烧录以下例程，熟悉 for 语句控制伺服电机的方法。



对应 C 语言代码为：

```
#include <Servo.h>

int _ABVAR_1_i = 0;
Servo servo_pin_3;
Servo servo_pin_4;

void setup()
{
  servo_pin_3.attach(3);
  servo_pin_4.attach(4);
}

void loop()
{
  for (_ABVAR_1_i = 1; _ABVAR_1_i <= (180); _ABVAR_1_i++)
  {
    servo_pin_3.write(_ABVAR_1_i);
    delay(20);
    servo_pin_4.write(_ABVAR_1_i);
    delay(20);
  }
}
```

3. 改写例程，让标准伺服归位的时候也能够缓慢运动。

4. 改进云台机构，使扫描范围更大。

实验 18 数字量传感器测值

实验目的：1. 了解数字量传感器和模拟量传感器的概念和区别；2. 学会利用串口监视器监测数字量传感器采集到的数值；3. 熟悉 serial 系列函数。

实验性质：验证型实验

实验课时：2 课时

实验步骤：

传感器可以大体上分为数字量传感器和模拟量传感器。我们之前在设置引脚值得时候接触过类似的概念，比如“设置引脚数字值”和“设置引脚模拟值”。传感器都可以从外界环境中检测到一些信号，数字量传感器检测到的信号是 0 或者 1（即“未触发”和“触发”），而模拟量传感器检测到的信号则是一个范围内的许多数值，这些数值都是电信号。

传感器的数值都可以通过一个叫做 serial monitor（串口监视器）的工具获取。我们只要编写一段代码就可以实现这个功能。

1. 数字量传感器可以用这种方法监测触发方式和触发条件，但是效果其实和“触发点亮 LED” 什么区别。我们可以用的语句是：



```
void setup()
{
  pinMode( 14, INPUT);
  Serial.begin(9600);
}

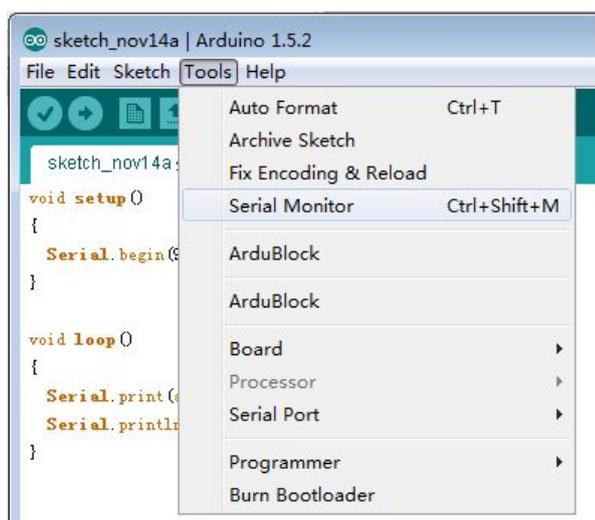
void loop()
{
  Serial.print(!( digitalRead(14) ));
  Serial.println();
}
```

2. 在“实用命令”菜单里可以找到“串口打印加回车”这个图形语句。“串口打印”意思就是在串口监视器里显示，对应的函数是 `Serial.print()`。我们把这个程序烧录之后，就可以接上黑标/白标传感器，并打开 serial monitor 查看检测到的数据。

在图形化界面的上方最右侧，有 serial monitor 按钮。



在 C 语言界面的 Tools 菜单里面，也可以找到 serial monitor 选项



另外，C 语言界面上方最右侧的“放大镜”按钮，也是 serial monitor。



打开之后即可看到传感器获取到的数据。

在监测过程中，主控板必须烧录上文提到的程序，且必须始终连在电脑上。

3. 搭建电路，分别监测触碰、触须、近红外、光强、黑标、声控这 6 种传感器。

串口监视器其实有很多，大家可以在网上搜索、下载各种版本的串口监视器，以开阔思路。

实验 19 模拟量传感器测值

实验目的：1. 了解传感器测值的概念；2. 学会利用 serial monitor 监测模拟量传感器的数值采集；3. 学会使用图形化程序编写测值程序；4. 了解数字量传感器和模拟量传感器的不同。

实验性质：验证型实验

实验课时：2 课时

实验步骤：

上节我们讲到，模拟量传感器检测到的数值是某个范围内的许多数值，那么这个数据是如何被测到的呢？

1. 我们仍然是通过 serial monitor（串口监视器）获取。请编写以下程序并烧录。



```
void setup()
{
  Serial.begin(9600);
}

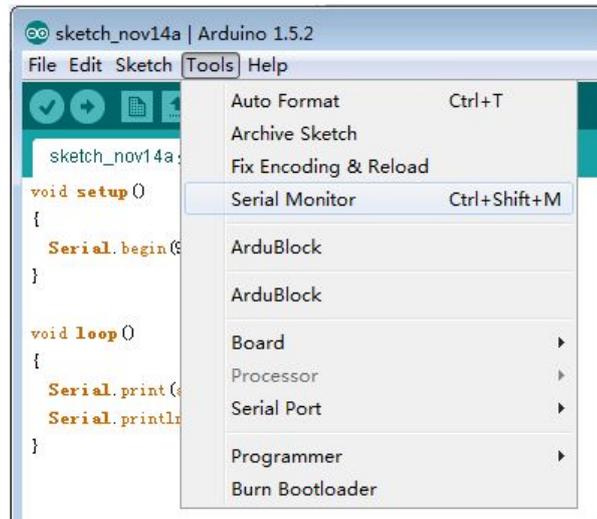
void loop()
{
  Serial.print(analogRead(14));
  Serial.println();
}
```

2. 在“实用命令”菜单里可以找到“串口打印加回车”这个图形语句。“串口打印”意思就是在串口监视器里显示，对应的函数是 Serial.print ()。我们把这个程序烧录之后，就可以接上黑标/白标传感器，并打开 serial monitor 查看检测到的数据。

在图形化界面的上方最右侧，有 serial monitor 按钮。



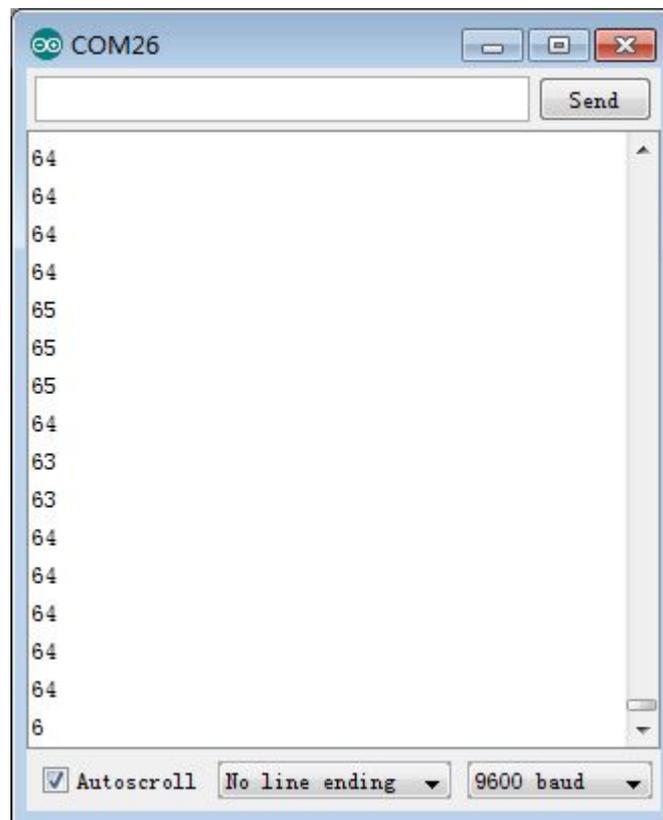
在 C 语言界面的 Tools 菜单里面，也可以找到 serial monitor 选项



另外，C 语言界面上方最右侧的“放大镜”按钮，也是 serial monitor。



打开之后即可看到传感器获取到的数据



在监测过程中，主控板必须烧录上文提到的程序，且必须始终连在电脑上。

学会了这个技能，我们就可以利用 serial monitor 去了解各种传感器的触发条件，以及检

测到的数据情况了。如果数据太乱，就可以加上延迟语句，隔一段时间监测一次。

3. 搭建电路，监测光强、黑标、声控这 3 种传感器。上个实验我们已经监测过这两个传感器了，为什么现在又用到了呢？因为这三种传感器同时具备数字量和模拟量的功能。

你可以使用手电筒、遮挡来测试光强；用不同深浅的色彩来测试黑标（此时黑标就有灰度的作用了）；自己发声来测试声控。

4. 那么其他的传感器呢？比如超声波、温湿度等等。这些传感器功能比较复杂，不能直接监测，后面我们会陆续讲到它们的监测方法。

实验 20 超声波传感器测距算法

实验目的：1. 练习用串口监视器监测超声波传感器；2. 理解超声测距传感器的；3. 理解超声测距算法的数学意义。

实验性质：验证型实验

实验课时：2 课时

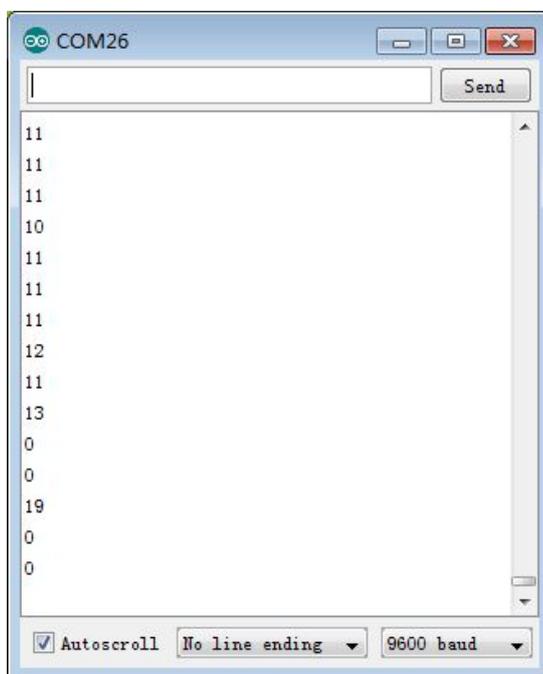
实验步骤：

1. 串口监视器测值

超声波有两个数据针脚，我们可以用到超声波专用的接口图块来完成监测



监测到数据如下：



如果我们拿一个尺子并在超声波传感器前面移动障碍物，我们会发现这个监测到的数据确实是比较准确的距离值，单位是 cm。

超声测距传感器是有图块的，而像温湿度、颜色识别、红外编码器等传感器没有专用的图块，直接读的话，读出来的数据又看不懂，这时候我们就需要研究一下这个传感器了。

2. 测距算法

超声波的例程中，查看生成的 C 语言，我们会发现源代码比较复杂：

```
int ardublockUltrasonicSensorCodeAutoGeneratedReturnCM(int trigPin, int
echoPin)
{
    long duration;
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(20);
    digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);
    duration = duration / 59;
    if ((duration < 2) || (duration > 300)) return false;
    return duration;
}

void setup()
{
    Serial.begin(9600);
    digitalWrite( 15 , LOW );
}

void loop()
{
    Serial.print(
ardublockUltrasonicSensorCodeAutoGeneratedReturnCM( 15 , 14 ));
    Serial.println();
}
```

以上这段 C 代码是自动生成的（其实是 Ardublock 的开发者写的）。

如果我们打开器材配套的资料，在电子模块资料，超声测距传感器的资料文件中可以发现，“探索者”的工程师也写了一个例程：

```
#define ECHOPIN 14
#define TRIGPIN 15
```

```

void setup()
{
  Serial.begin(9600);
  pinMode(ECHOPIN, INPUT);
  pinMode(TRIGPIN, OUTPUT);
}

void loop()
{
  digitalWrite(TRIGPIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGPIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGPIN, LOW);
  float distance = pulseIn(ECHOPIN, HIGH);
  distance= distance/58;
  Serial.println(distance);
  delay(500);
}

```

这两段代码功能是一样的，大家要对比阅读。它们的思路是一样的，在数据处理上细节有些小差异，比如在计算距离数值时，一个是拿 59 做除数，一个是拿 58 做除数。这两段代码无所谓哪个更权威，代码都是普通人写的，所以用户不要迷信“例程”，要根据自己的需要，对代码进行选择使用或者改写。

里面有一个陌生的函数 pulseIn()，我们可以上网查询这个函数的功能，由于 Arduino 是一个国际通用的开源体系，所以用户量非常庞大，互联网上有海量的共享资源和交流人群，所以一定能搜索到。

经过查询我们可以知道 pulseIn()的功能是获取两个信号的时间差，即发出超声波到收到反射回来的超声波的时间差，单位是微秒。有了这个时间差，那么中学物理知识就派上用场了：

$$\text{距离 } D = \text{声速 } v \times \text{往返时间差 } t/2$$

根据中学物理知识，我们知道声波在空气中的传播速度是 340m/s，但是这个数据是有前提的：1 个标准大气压和 15°C 的条件下。经再次搜索可知，超声波在 1 个标准大气压和

20°C的条件下速度为 344m/s。如果改成 cm/μs 为单位，那么

$$D=344 \times 10^2 \times 10^{-6} \times t/2=0.0172t \approx t/58$$

所以超声波测距算法约定俗成的写作：

```
distance = pulseIn(ECHOPIN, HIGH);  
distance= distance/58;
```

3. 知道了这些原理，我们就可以写出超声波测距功能程序。假设我们选择一个直流电机，设定这样一个功能：当目标离它 10cm 以下时，不动，超过 10cm 时，开始转动，那么程序可以写作：



看懂了这个程序后，请大家改写这个程序，用 023 号机构或其他机构实现超声避障或跟随功能。

4. 这样，我们就从原理到实践，完全掌握了一个陌生的，比较复杂的传感器。

```
从上面的案例中，我们大致可以总结出一套通用的学习传感器的方法：  
  
(1) 当我们遇到一个新的传感器时，首先阅读它的例程和资料，从而知道它的工作原理、信号输出引脚、信号类型、涉及到的语句或函数，有不清楚的就去查询；
```

(2) 进而通过串口监视器监测它的工作状态，验证它的工作方式和数据生成情况等。

(3) 根据它的工作方式，将它安排在机身的合适位置，根据它的数据生成情况，编写合适的程序语句。

阅读例程和学会看电路(能看针脚功能就行)是非常重要的。能够熟练地查询相关资料也是很重要的。现在是信息爆炸的时代,这种开源硬件、常用电子模块和编程的资料、教程、案例等都非常容易查到。用户要尽量自己查询,自己摸索,实在解决不了可以找技术人员询问**参考资料**的地址,切莫做伸手党——直接要答案。

“探索者”器材里涉及到的所有较复杂的传感器,如超声测距、温湿度、红外编码器、都给出了串口监视例程。请大家学习的时候自己查询使用。

实验 21 超声雷达实验

实验目的：1. 掌握超声测距传感器的一种应用；2. 进一步练习编程。

实验性质：设计型实验

实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板，Bigfish 扩展板；Bigfish 连接常规传感器.ppt；实验 20；

主要器材：超声波传感器×1；023 号机构×1；Led×1

实验步骤：

1. 组装一台搭载超声波传感器的智能车，连接好电路。
2. 阅读。编写并烧录如下程序：该段代码将实现如下功能，当小车倒车时，如若后方有障碍物，则 LED 灯的闪烁频率和距离成反比。即距离越小，闪烁频率越高，以起到警示作用。

```
#define antenna1 A0
#define antenna2 A4 //宏定义 2 个触须传感器引脚
#define led0 A4
#define led1 A5
int speed1=150;
int speed2;
int distance;
#define zero 0 //定义 3 个速度
int Echo = A1; // Echo 回声脚(P2.0)
int Trig =A0; // Trig 触发脚(P2.1)
unsigned long time;
void March(); //Sir,yeah Sir 前进
void Fallback(); //敌方火力太猛，后退
void Left(); //左转
void Right(); //右转
void Attention(); //稍息(就是停下来啦！)
int a;
void setup() {
pinMode(antenna1,INPUT);
pinMode(antenna2,INPUT);
pinMode(5,OUTPUT);
pinMode(6,OUTPUT);
pinMode(9,OUTPUT);
pinMode(10,OUTPUT); //初始化引脚
pinMode(Echo, INPUT); // 定义超声波输入脚
pinMode(Trig, OUTPUT); // 定义超声波输出脚
}
int Distance_test() // 量出前方距离
{
digitalWrite(Trig, LOW); // 给触发脚低电平 2μs
delayMicroseconds(2);
digitalWrite(Trig, HIGH); // 给触发脚高电平 10μs，这里至少是 10μs
delayMicroseconds(20);
```

```

digitalWrite(Trig, LOW); // 持续给触发脚低电
float Fdistance = pulseIn(Echo, HIGH); // 读取高电平时间(单位：微秒)
Fdistance= Fdistance/58; //为什么除以 58 等于厘米， Y 米= (X 秒*344 )
/2
// X 秒= ( 2*Y 米 ) /344 ==》 X 秒=0.0058*Y 米 ==》 厘米=微秒/58
return (int)Fdistance;
}
void flash(){
digitalWrite(led0,HIGH);
digitalWrite(led1,LOW);
delay(a);
digitalWrite(led0,HIGH);
digitalWrite(led1,LOW);
delay(a);
}
void loop() {
distance = Distance_test();
if(distance>=35)
{a=1000;flash();}
if(distance<=35)
{a=1000-pow((35-distance),2);flash();}
}

```

3. 思考程序，如何修改 LED 灯的闪烁频率？

实验 22 机械手爪模块

实验目的 :1. 熟悉一种机械手模块的结构特点和组装规律 ;2. 学会利用小型标准伺服电机、齿轮组等组装机械手爪模块 ;3. 熟悉机械手爪模块在机器人结构设计中的应用。

实验性质 :验证型实验

实验课时 :2 课时

参考资料 :000_module006a_rh1s.jpg ;标准舵机的安装.ppt ;Bigfish 连接伺服电机.ppt ;

实验 11。

实验步骤 :

1. 组装一个机械手

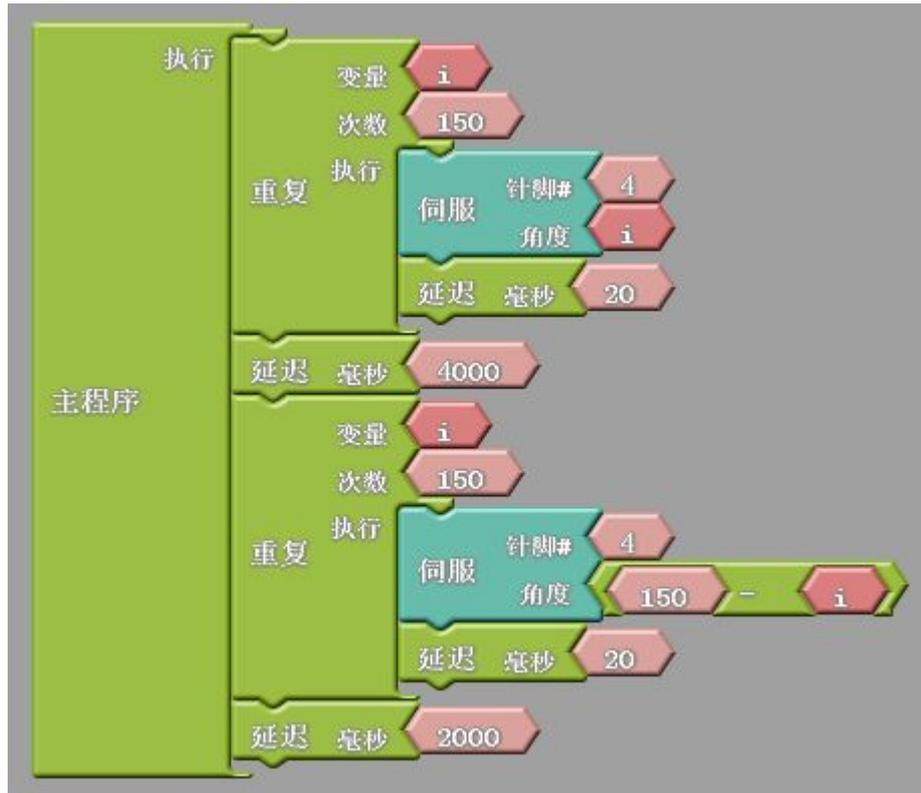


图 16.1 机械手模块 module006a

结构说明 :机械手模块 module006a ,由 1 个标准伺服电机驱动 ,通过连杆结构和齿轮组传动来达到夹取效果 ,机械手的设计方案有很多种 ,这只是其中一种。

运动特性 :开合角度比较大 ,夹具顶端的运动轨迹简单稳定 ,夹具顶端非平行开合 ,比较适合于“握”住曲面物体或者柔软的物体。夹具顶点可以再安装一对带铰接的小平板零件 ,从而适合夹取立方体形状的物体。

2. 将 Mehran 主控板、Bigfish 扩展板、锂电池和关节模块连接成电路 ;由于也是标准舵机 ,因此可以参考实验 11、12 ,在图形化编程界面 Ardublock 中编写以下程序并烧录 :



意思是：机械手爪慢慢合拢，合拢后保持 4 秒，然后慢慢张开，张开保持 2 秒；循环。

C 语言代码为：

```

#include <Servo.h>

int _ABVAR_1_i = 0;
Servo servo_pin_4;

void setup()
{
  servo_pin_4.attach(4);
}

void loop()
{
  for (_ABVAR_1_i = 1; _ABVAR_1_i <= ( 150 ); _ABVAR_1_i++)
  {
    servo_pin_4.write( _ABVAR_1_i );
    delay( 20 );
  }
  delay( 4000 );
  for (_ABVAR_1_i = 1; _ABVAR_1_i <= ( 150 ); _ABVAR_1_i++)
  {
    servo_pin_4.write( ( 150 - _ABVAR_1_i ) );
    delay( 20 );
  }
  delay( 2000 );
}

```

3. 由于图形化不能做递减运算，所以我们采用了“150-i”这个计算方法。我们也可以

用递减运算重写上面的程序。

```
#include <Servo.h>

int i;
int j;
Servo servo_pin_4;

void setup()
{
  servo_pin_4.attach(4);
}

void loop()
{
  for (i= 1; i<= 150;i++ )
  {
    servo_pin_4.write(i);
    delay( 20 );
  }
  delay( 4000 );
  for (j= 150; j>= 0 ;j-- )
  {
    servo_pin_4.write( j);
    delay( 20 );
  }
  delay( 2000 );
}
```

实验 23 3 自由度机械臂

实验目的：1. 熟悉 3 自由度机械臂的结构特点和组装规律；2. 学会利用关节模块、机械手爪模块组装串联工业机器人；3. 熟悉标准舵机对应的函数，尝试运动规划。

实验性质：设计型实验

实验课时：2 课时

参考资料：230_3df robot_3s.stp；实验 12;实验 16；

实验步骤：

1. 组装一个 230 号结构

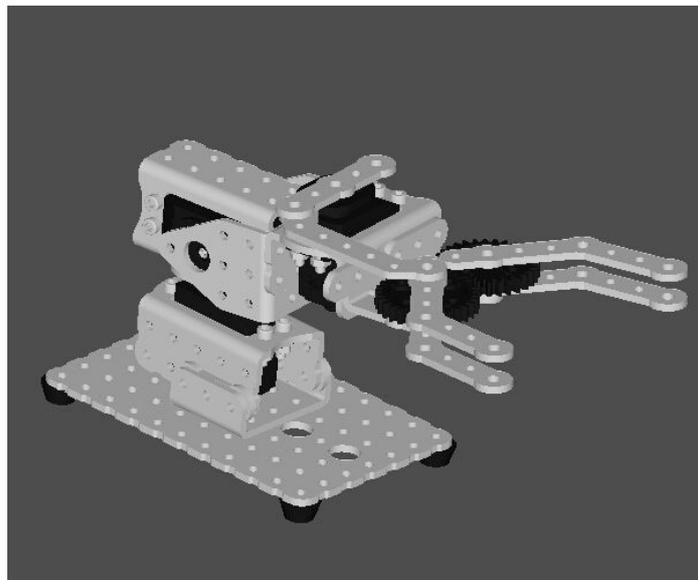


图 17.1 三自由度机械手

2. 参考 2 自由度云台和机械手爪的控制方法，编写一段程序，控制机械臂模拟工作运动。
3. 尝试改造该机构的关节串联方向，形成一个新机构，并控制。

实验 24 按颜色分拣

实验目的：1. 5 自由度机械臂多种颜色识别分拣；2. 掌握伺服电机的工作特性；3. 了解黑标传感器的工作方法及程序设计。

实验性质：设计型实验

实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板，Bigfish 扩展板；Bigfish 连接常规传感器.ppt；实验 23；

主要器材：3 自由度机械臂×1；灰度传感器×1；

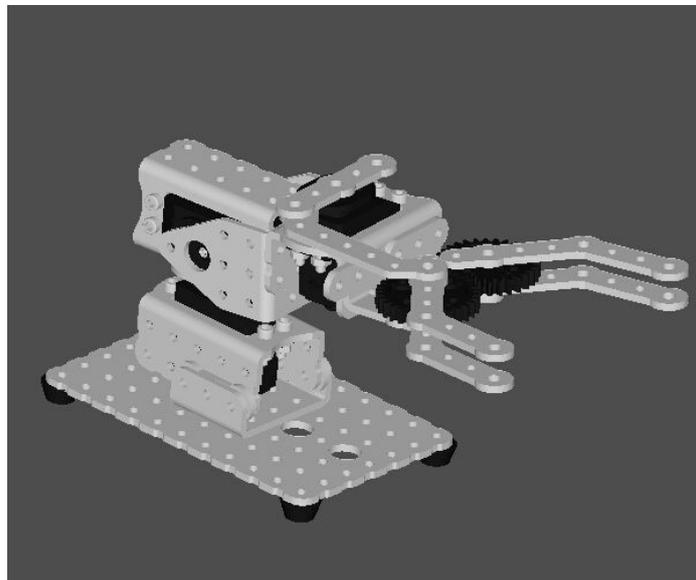


图 17.1 三自由度机械手

实验步骤：

1. 在主控板上搭载黑标传感器，连接好电路。
2. 阅读、编写并烧录如下程序：

```
#define shar A0
void setup() {
  pinMode(shar,INPUT);
  Serial.begin(9600);
}

void loop() {
```

```
int a= analogRead(shar);
int c= map(a,0,1000,0,10);
if(a<=5)
  Serial.print("black");
else
  Serial.pring("white");
}
```

3. 代码实现的功能为：识别物体的灰度值，当用黑白两种物体分别置于灰度传感器前，在串口处打印出物体的颜色为 black 或 white。
4. 组装一台 3 自由度机械臂，搭载黑标传感器，连接好电路。参考实验 23 设计程序，使机械臂可以识别黑白两种物料，并使机械臂抓取不同颜色的物料放置在指定点。

实验 25 有限状态机算法实验

实验目的：1. 了解一种循迹功能的实现方式；2. 了解灰度传感器的工作特性及安装注意事

项。3. 掌握 switch 语句的写法。4. 了解“有限状态机”的概念，掌握一种常用的编程框架，学会分析传感器触发组合状态。

实验性质：验证型实验

实验课时：4 课时

参考资料：请参考 Bigfish 连接常规传感器.ppt，实验 13；“探索者”编程框架.ppt

主要器材：灰度传感器×3

有限状态机 (Finite-state machine) 简称 FSM，表示有限个状态以及在这些状态之间的转移和动作等行为的数学模型。它把复杂的控制逻辑分解成有限个稳定状态，在每个状态上判断事件。由于有限状态机有有限个状态，因此可以在实际中实现。有限状态机可以广泛的应用于机器人多个传感器触发组合状态的判断，大大提高检测效率。

状态表

之前我们了解过条件判断，机器人的传感器触发一般用条件判断来做。

这时机器人程序的一般思路是：

```
如果
机器人的某几个传感器触发了；
机器人的某几个电机做个什么事；
做多久；
如果
机器人的另外某几个传感器触发了；
机器人的某几个电机做个什么事；
做多久；
```

所以我们总是要用到大量的 if 语句。比如双轮小车的某个功能：

```
如果
机器人的 1 号传感器触发了；
机器人的左侧电机顺时针转；
机器人的右侧电机逆时针转；
持续 5 秒；
如果
```

机器人的 2 号传感器触发了；
机器人的左侧电机逆时针转；
机器人的右侧电机顺时针转；
持续 5 秒；
否则
都不转

用伪码写出来就是：

```
if { Sensor(端口 a,触发); //传感器触发时此句为真，否则为假 }  
{  
    Motor(L,顺);  
    Motor(R,逆);  
    Delay 5;  
}  
if { Sensor(端口 b,触发); }  
{  
    Motor(L,逆);  
    Motor(R,顺);  
    Delay 5;  
}  
else  
{  
    Motor(L,停);  
    Motor(R,停);  
}
```

无论是 ARM7 还是 Mehran 都很容易实现。

在只有一个传感器的情况下，我们假设这是个数字量传感器。那么我们可以得到一个状态表格：

状态序号	传感器 1
1	1
2	0

这个传感器有两个状态。

而当有两个传感器时，则有四个状态。

状态序号	传感器 1	传感器 2
1	1	1

2	1	0
3	0	1
4	0	0

如果我们用 if 语句写这四个状态，就显得比较长。

状态序号	传感器 1	传感器 2	伪码
1	1	1	if { Sensor(1,1); Sensor(2,1); }
2	1	0	if { Sensor(1,1); Sensor(2,0); }
3	0	1	if { Sensor(1,0); Sensor(2,1); }
4	0	0	else

在编程的时候，状态罗列的越全，未来机器人的 bug 越少。但是随着传感器的增多，状态数量按 2 的 N 次幂增加，大量的 if 语句使执行效率变得很低，经常出现识别不灵的情况。我们有必要换一种高效写法。

多个确定数量的传感器的触发组合，符合有限状态机的概念，有限状态机一般是用 Switch 语句来实现。如：

```

switch(s)
{
case 1 : {动作 1;}break;
case 2 : {动作 2;}break;
case 3 : {动作 3;}break;
case 4 : Act_Stop();break;
default;;break;
}

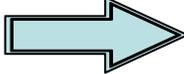
```

不难发现，这段语句实现的关键，就是识别出上页表中的 1、2、3、4，四个状态序号。

那么问题就来了：我们如何让机器人知道自己传感器的触发组合对应于 1、2、3、4 的哪个序号呢？

二进制状态表

下面，我们把每组传感器返回值看成一个二进制数值。

传感器 1	传感器 2		二进制结果	十进制结果
1	1		11	3
1	0		10	2
0	1		01	1
0	0		00	0

结果我们发现了一种新的、可计算的编码方式：

新序号	传感器 1	传感器 2
0	0	0
1	0	1
2	1	0
3	1	1

于是，只要我们知道了传感器们的触发状态，也就知道了序号；知道了序号，也就知道了传感器们的触发状态。用这个序号去写 switch 语句，再合适不过了。下面我们要做的是，用一种算法，让机器人能够返回自己接收到的传感器组合值的二进制数据。

算法精解

我们可以使用以下算法来实现：

- 首先设置一个变量 s ，这个 s ，将存储传感器组的二进制状态序号。
- 我们还需要用到一个重要的运算符 “ \ll ”，这个运算符的意义是：左移
如： $1 \ll n$ ，意思是 1 向左移动 n 位，空出来的数位用 0 填补。

如： $1 \ll 1$ ，结果就是 10； $1 \ll 2$ ，结果就是 100； $101 \ll 1$ ，结果就是 1010

- 只要让机器人依次返回各个传感器的状态数值，最早获取的，移到最左；第二获得的，移到“倒数第二左”，……，以此类推。即可获得。

如两个传感器均触发：

先获得 1 号的数值（真）并左移 0 位，得

0	1
---	---

再获得 2 号的数值（真）并左移 1 位，得

1	0
---	---

两数值取“或”，即可得 11

数学问题解决了，很容易就可以转化为程序语句：

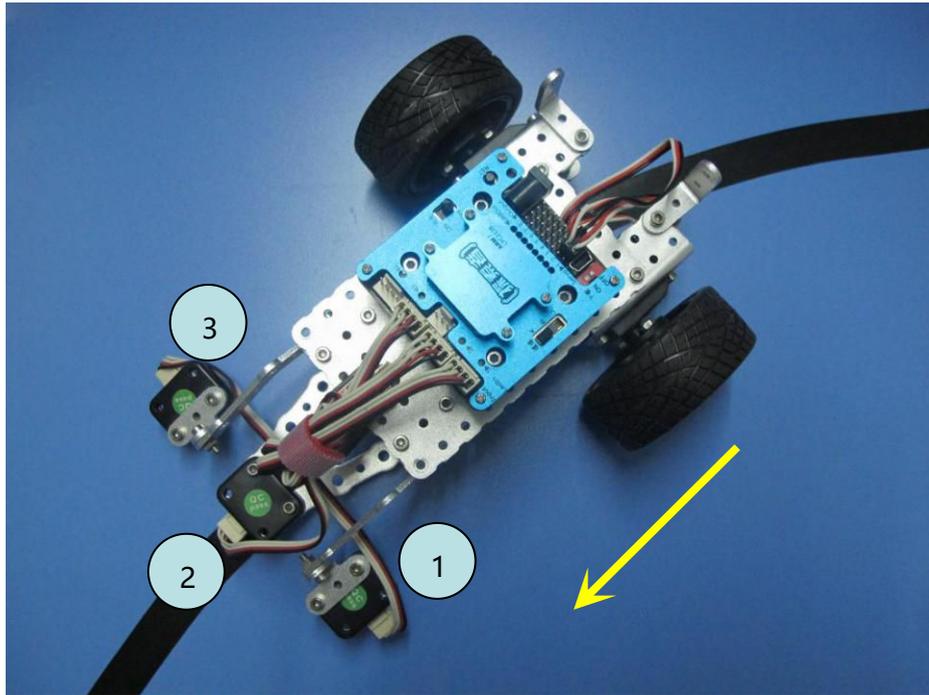
```
s=0;
for(i=0;i<2;i++) //因为此例中有 2 个传感器，i 取 2
{
s=s|(Servo(i+1, 触发判断)<<i); //获得传感器值，移位，或运算
}
```

于是 switch 语句可以写为：

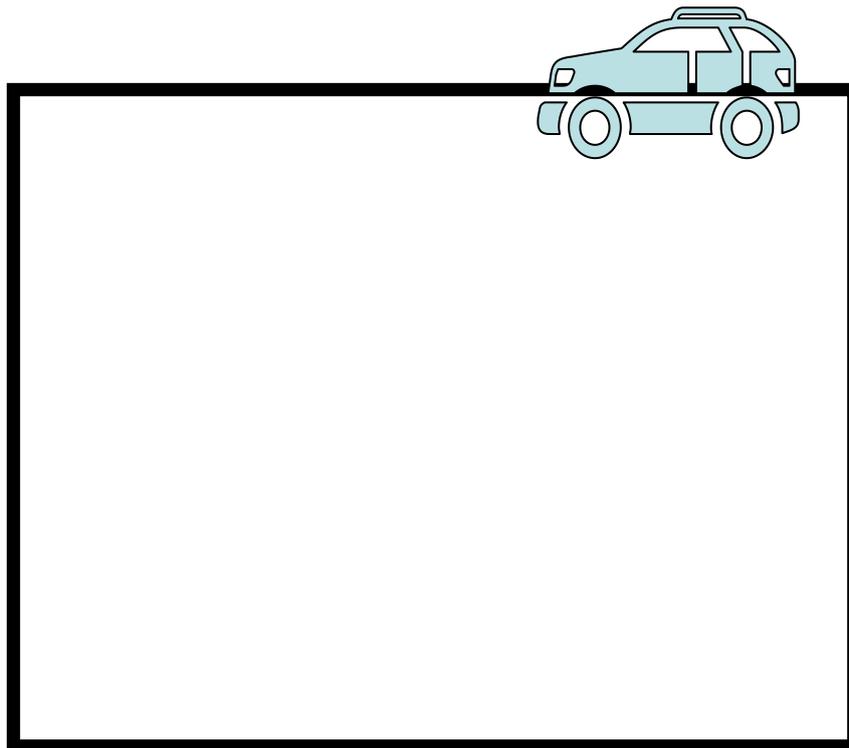
```
switch(s)
{
case 0x00 : {动作 0;}break; //序号也可以写作 16 进制数值
case 0x01 : {动作 1;}break;
case 0x02 : {动作 2;}break;
case 0x03 : {动作 3;}break;
default;:break;
}
```

策略表

下面我们以一个“三传感器双轮循迹小车”程序为例，再来推导一遍：



假设跑道是矩形，行进方向如图：



传感器触发情况、小车行驶状态、对应行为策略表如下：

传感器 1	传感器 2	传感器 3	序号	小车状态	动作
0	0	0	0	都没触发，可能是跑偏了	停
0	0	1	1	小车左偏	左轮逆时针转，向右调整
0	1	0	2	小车正中	左轮逆时针转，右轮顺时针转，前进
0	1	1	3	在这个行进方向上不可能	无
1	0	0	4	小车右偏	右轮顺时针转，向左调整
1	0	1	5	在此跑道上不可能	无
1	1	0	6	遇到转角	右轮顺时针转，左转
1	1	1	7	在此跑道上不可能	无

伪码如下：

```

s=0;
for(i=0;i<3;i++)
{
s=s|(Input(i+1,1)<<i);
}
switch(s)
{
case 0x00 : 停;break;
case 0x01 : {Motor(L,逆);Motor(R,停);}break;
case 0x02 : {Motor(L,逆);Motor(R,顺);}break;
case 0x04 : {Motor(L,停);Motor(R,顺);}break;
case 0x06 : {Motor(L,停);Motor(R,顺);}break;
default;;break;
}

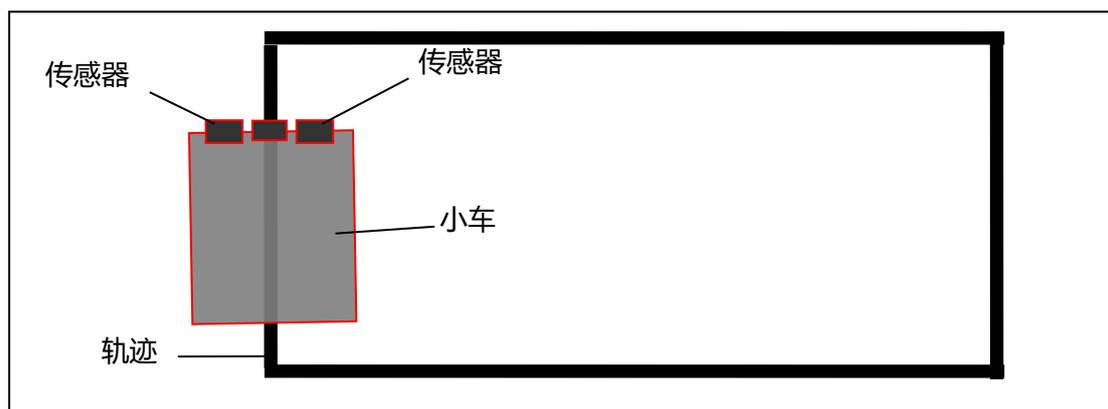
```

这段代码中的动作，完全由策略表分析获得，因此，当状态比较多时，用户要学会利用策略表进行分析，从而确定机器人的动作策略，而不是凭空想象。

实验步骤：

1. 将 3 个灰度传感器安装于 023 机构的底部前端，连接好电路；
2. 实验 9 中，我们学习了一种通过条件判断来进行循迹的编程方法，这种方法思路简单，

实现快，但是循迹效果一般，经常出现没有正常识别到黑线的情况。主要原因是 if 语句是比较基础的程序语句，一旦计算量变大，判断条件变多，难度变大，如“瞬时对多个传感器状态进行同步判断”时，就不太够用了。那么我们就需要用另外一种思路来编程。请大家阅读“探索者编程框架.ppt”，学习以下程序，根据自己的机构实际情况调整参数，实现稳定的 3 传感器巡线。该段程序为，有 90 度转角的，黑色轨迹上的，3 传感器巡线。



为了让程序结构更规范，我们可以为小车编写子函数，大家可以根据下面的子函数的写法来调整电机的接线，务必让子函数的定义和小车现实中运动是相符的。

左转：

```
void Left()
{
  digitalWrite( 5, HIGH );
  digitalWrite( 6, LOW );
  digitalWrite( 9, LOW );
  digitalWrite( 10, LOW );
}
```

右转：

```
void Right()
{
  digitalWrite( 5, LOW );
  digitalWrite( 6, LOW );
  digitalWrite( 9, HIGH );
  digitalWrite( 10, LOW );
}
```

前进：

```
void Forwards()
{
```

```

digitalWrite( 5 , HIGH );
digitalWrite( 6 , LOW );
digitalWrite( 9 , HIGH );
digitalWrite( 10 , LOW );
}

```

停止：

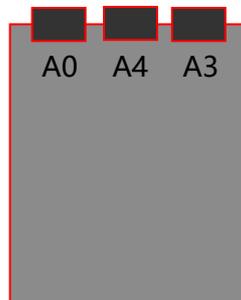
```

void Stop()
{
digitalWrite( 5 , LOW );
digitalWrite( 6 , LOW );
digitalWrite( 9 , LOW );
digitalWrite( 10 , LOW );
}

```

电路连接：当小车车顶如下图所示对着你时，从左到右的传感器分别连接 A0、A4、A3 口。

电机的连接方式请大家自己摸索。



完整程序为：

```

int pin[3] = {A0, A4, A3};
byte value;
byte value_his = 0; //记录上一次的传感器值
void setup()
{
pinMode( 5 , OUTPUT);
pinMode( 6 , OUTPUT);
pinMode( 9 , OUTPUT);
pinMode( 10 , OUTPUT);
}
void loop()
{
value = 0;
for(int i=0; i<3; i++)
{
value |= (digitalRead(pin[i]) << i);
}
if(value == 0x07){ //当传感器都没有触发时默认为上一次的值
value = value_his;
}
switch (value)
{
case 0x00: //全部触发
Forwards();
break;
case 0x01: //触发右边两个
while(digitalRead(pin[1])){ //通过 while 循环使小车回到跑道中间

```

```

Right();
}
break;
case 0x03: //触发右边一个
while(digitalRead(pin[1]))
{
Right();
}
break;
case 0x04: //触发左边两个
while(digitalRead(pin[1]))
{
Left();
}
break;
case 0x05: //触发中间一个
Forwards();
break;
case 0x06: //触发左边一个
while(digitalRead(pin[1]))
{
Left();
}
break;
default:
Stop();
}
value_his = value;
}

void Left()
{
digitalWrite( 5 , HIGH );
digitalWrite( 6 , LOW );
digitalWrite( 9 , LOW );
digitalWrite( 10 , LOW );
}

void Right()
{
digitalWrite( 5 , LOW );
digitalWrite( 6 , LOW );
digitalWrite( 9 , HIGH );
digitalWrite( 10 , LOW );
}

void Forwards()
{
digitalWrite( 5 , HIGH );
digitalWrite( 6 , LOW );
digitalWrite( 9 , HIGH );
digitalWrite( 10 , LOW );
}

void Stop()
{
digitalWrite( 5 , LOW );
digitalWrite( 6 , LOW );
}

```

```
digitalWrite( 9 , LOW );  
digitalWrite( 10 , LOW );  
}
```

实验 26 LED 点阵笑脸

实验目的：1. 8×8LED 点阵的特性；2. 学会调用点阵字库；3. 学会使用字库显示需要的图案。

实验性质：设计型实验

实验课时：2 课时

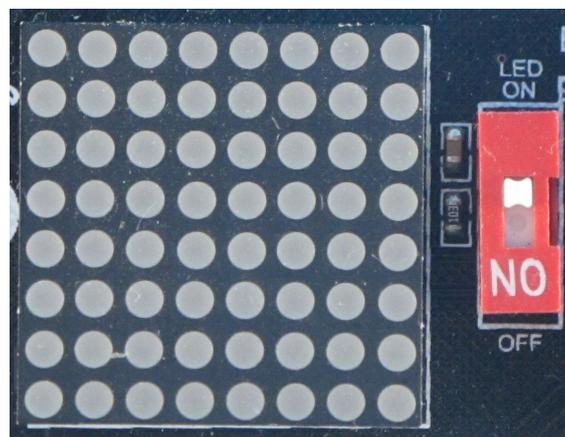
参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板，Bigfish 扩展板；Bigfish 连接常规传感器.ppt；Arduino Reference；

主要器材：Bigfish 扩展版

实验步骤：

8×8LED 点阵位于 Bigfish 扩展板上，可用于机器人设备工作过程中的状态显示或信息显示，如显示数字、字母、符号等，或用来制作简单的电子项目，如电子游戏机等。使用前需要安装扩展库。

8*8led 模块采用 MAX7219 驱动芯片，有端口与部分舵机复用，因此点阵和部分舵机不能同时使用。



实验目的：熟悉 LED 点阵，学会安装 LED 点阵的扩展库；学会使用 LED 点阵显示信息。

实验性质：验证型实验

实验课时：2 课时

实验步骤：

将 LED 扩展库 `电子元件\控制\BigFish 扩展板\libraries\LedControl` 拷贝到 arduino 的 libraries 中。

启动 arduino , 可以发现 , 在 Example 中增加了 MultiLCD 的例程 , 打开
File-Examples-LedControl-LCDemoMatrix

将例程 uploading 到控制板中 , 观察 Led 点阵上显示的内容。

```
#include "LedControl.h"
LedControl lc=LedControl(12,11,13,1);

unsigned long delaytime=100;

void setup() {
  lc.shutdown(0,false);
  lc.setIntensity(0,8);
  lc.clearDisplay(0);
}

void writeArduinoOnMatrix() {
  /* here is the data for the characters */
  byte a[5]={B01111110,B10001000,B10001000,B10001000,B01111110};
  byte r[5]={B00111110,B00010000,B00100000,B00100000,B00010000};
  byte d[5]={B00011100,B00100010,B00100010,B00010010,B11111110};
  byte u[5]={B00111100,B00000010,B00000010,B00000100,B00111110};
  byte i[5]={B00000000,B00100010,B10111110,B00000010,B00000000};
  byte n[5]={B00111110,B00010000,B00100000,B00100000,B00011110};
  byte o[5]={B00011100,B00100010,B00100010,B00100010,B00011100};

  lc.setRow(0,0,a[0]);
  lc.setRow(0,1,a[1]);
  lc.setRow(0,2,a[2]);
  lc.setRow(0,3,a[3]);
  lc.setRow(0,4,a[4]);
  delay(delaytime);
  lc.setRow(0,0,r[0]);
  lc.setRow(0,1,r[1]);
  lc.setRow(0,2,r[2]);
  lc.setRow(0,3,r[3]);
  lc.setRow(0,4,r[4]);
  delay(delaytime);
  lc.setRow(0,0,d[0]);
  lc.setRow(0,1,d[1]);
  lc.setRow(0,2,d[2]);
  lc.setRow(0,3,d[3]);
  lc.setRow(0,4,d[4]);
  delay(delaytime);
  lc.setRow(0,0,u[0]);
  lc.setRow(0,1,u[1]);
  lc.setRow(0,2,u[2]);
  lc.setRow(0,3,u[3]);
  lc.setRow(0,4,u[4]);
  delay(delaytime);
}
```

```

lc.setRow(0,0,i[0]);
lc.setRow(0,1,i[1]);
lc.setRow(0,2,i[2]);
lc.setRow(0,3,i[3]);
lc.setRow(0,4,i[4]);
delay(delaytime);
lc.setRow(0,0,n[0]);
lc.setRow(0,1,n[1]);
lc.setRow(0,2,n[2]);
lc.setRow(0,3,n[3]);
lc.setRow(0,4,n[4]);
delay(delaytime);
lc.setRow(0,0,o[0]);
lc.setRow(0,1,o[1]);
lc.setRow(0,2,o[2]);
lc.setRow(0,3,o[3]);
lc.setRow(0,4,o[4]);
delay(delaytime);
lc.setRow(0,0,0);
lc.setRow(0,1,0);
lc.setRow(0,2,0);
lc.setRow(0,3,0);
lc.setRow(0,4,0);
delay(delaytime);
}

void rows() {
    for(int row=0;row<8;row++) {
        delay(delaytime);
        lc.setRow(0,row,B10100000);
        delay(delaytime);
        lc.setRow(0,row,(byte)0);
        for(int i=0;i<row;i++) {
            delay(delaytime);
            lc.setRow(0,row,B10100000);
            delay(delaytime);
            lc.setRow(0,row,(byte)0);
        }
    }
}

void columns() {
    for(int col=0;col<8;col++) {
        delay(delaytime);
        lc.setColumn(0,col,B10100000);
        delay(delaytime);
        lc.setColumn(0,col,(byte)0);
        for(int i=0;i<col;i++) {
            delay(delaytime);

```

```

        lc.setColumn(0,col,B10100000);
        delay(delaytime);
        lc.setColumn(0,col,(byte)0);
    }
}
}

void single() {
    for(int row=0;row<8;row++) {
        for(int col=0;col<8;col++) {
            delay(delaytime);
            lc.setLed(0,row,col,true);
            delay(delaytime);
            for(int i=0;i<col;i++) {
                lc.setLed(0,row,col,false);
                delay(delaytime);
                lc.setLed(0,row,col,true);
                delay(delaytime);
            }
        }
    }
}

void loop() {
    writeArduinoOnMatrix();
    rows();
    columns();
    single();
}

```

2. 阅读并理解 LED 点阵显示的规律，尝试改变其中的设置来实现图形的变化。
3. 阅读并烧录以下程序，在 LED 点阵上显示一个笑脸。

```

#include "LedControl.h"

LedControl lc=LedControl(12,11,13,1);

void setup()
{
    lc.shutdown(0,false);
    lc.setIntensity(0,8);
    lc.clearDisplay(0);

    lc.setRow(0,0,0xff);
    lc.setRow(0,1,B10000001);
    lc.setRow(0,2,B10100101);
    lc.setRow(0,3,B10000001);
    lc.setRow(0,4,B11000011);
    lc.setRow(0,5,B10100101);
    lc.setRow(0,6,B10011001);
    lc.setRow(0,7,0xff);
}

```

```
}
```

```
void loop()
```

```
{
```

```
}
```

实验 27 蓝牙电子温度计实验

实验目的：1. 了解温湿度传感器的特性；2. 了解蓝牙串口模块的特性；3. 学会使用蓝牙通信传递串口数据。4. 熟悉物联网的相关知识。

实验性质：设计型实验

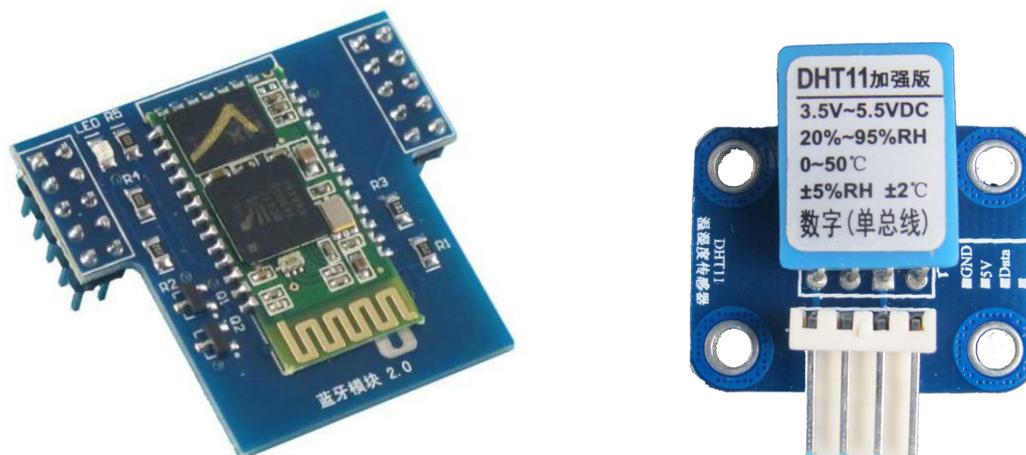
实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板，Bigfish 扩展板；Bigfish 连接常规传感器.ppt；Arduino Reference；

主要器材：Bigfish 扩展版

实验步骤：

1. 在 Basra 主控板上连接温湿度传感器，将蓝牙模块插在 Bigfish 扩展板上，连接好电路。在手机安装“蓝牙串口助手.apk”。下载安装后打开软件连接蓝牙。



2. 阅读编写并烧录以下程序：该段代码将实现远程蓝牙实时远程监控温湿度情况，并将数据反馈至手机端。

```
//摄氏温度转化为华氏温度
double Fahrenheit(double celsius){
    return 1.8 * celsius + 32;
}

//摄氏温度转化为开氏温度
double Kelvin(double celsius){
    return celsius + 273.15;
}
```

```

// 露点 (点在此温度时, 空气饱和并产生露珠)
// reference: http://wahiduddin.net/calc/density\_algorithms.htm
double dewPoint(double celsius, double humidity){
    double RATIO = 373.15 / (273.15 + celsius); // RATIO was originally
named A0, possibly confusing in Arduino context
    double SUM = -7.90298 * (RATIO - 1);
    SUM += 5.02808 * log10(RATIO);
    SUM += -1.3816e-7 * (pow(10, (11.344 * (1 - 1/RATIO)))) - 1);
    SUM += 8.1328e-3 * (pow(10, (-3.49149 * (RATIO - 1)))) - 1);
    SUM += log10(1013.246);
    double VP = pow(10, SUM - 3) * humidity;
    double T = log(VP/0.61078); // temp var
    return (241.88 * T) / (17.558 - T);
}

// 快速计算露点, 速度是 5 倍 dewPoint()
// reference: http://en.wikipedia.org/wiki/Dew\_point
double dewPointFast(double celsius, double humidity){
    double a = 17.271;
    double b = 237.7;
    double temp = (a * celsius) / (b + celsius) + log(humidity/100);
    double Td = (b * temp) / (a - temp);
    return Td;
}

#include <DHT11.h>
dht11 DHT11;
#define DHT11PIN A0

void setup(){
    Serial.begin(9600);
    Serial.println("DHT11 TEST PROGRAM ");
    Serial.print("LIBRARY VERSION: ");
    Serial.println(DHT11LIB_VERSION);
    Serial.println();
}

void loop(){
    Serial.println("\n");
    int chk = DHT11.read(DHT11PIN);
    Serial.print("Read sensor: ");
    switch (chk)
    {
        case DHTLIB_OK:
            Serial.println("OK"); break;

```

```

    case DHTLIB_ERROR_CHECKSUM:
        Serial.println("Checksum error"); break;
    case DHTLIB_ERROR_TIMEOUT:
        Serial.println("Time out error"); break;
    default:
        Serial.println("Unknown error");break;
}

Serial.print("Humidity (%): ");
Serial.println((float)DHT11.humidity, 2);          //湿度

Serial.print("Temperature (oC): ");                //摄氏温度
Serial.println((float)DHT11.temperature, 2);

Serial.print("Temperature (oF): ");                //华氏温度
Serial.println(Fahrenheit(DHT11.temperature), 2);

Serial.print("Temperature (K): ");                 //K 氏温度
Serial.println(Kelvin(DHT11.temperature), 2);

Serial.print("Dew Point (oC): ");                  //露点温度 (露珠凝结的温度？是的)
Serial.println(dewPoint(DHT11.temperature, DHT11.humidity));

Serial.print("Dew PointFast (oC): ");
Serial.println(dewPointFast(DHT11.temperature, DHT11.humidity));

delay(2000);
}

```

3. 也许有的同学下载别的蓝牙串口 App，接收到的数据大多是乱码，这是为什么呢？在程序中加入延迟环节会不会解决问题？

实验 28 速度协调实验

实验目的：1. 使用红外编码器实现速度反馈；2. 进一步练习编程。

实验性质：设计型实验

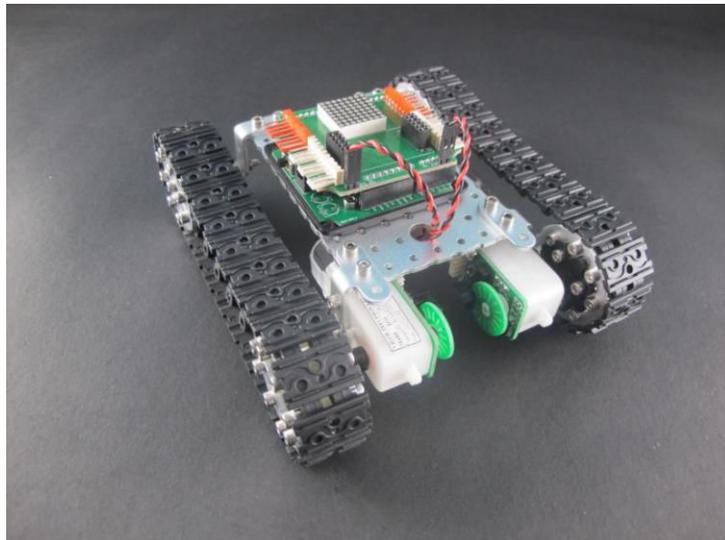
实验课时：2 课时

参考资料：请参考光盘中的“电子元件资料”，Basra 控制板，Bigfish 扩展板；Bigfish 连接常规传感器.ppt；实验 1；

主要器材：红外编码器×1

实验步骤：

1. 将红外编码器连接在履带小车一侧的直流电机上，连接好电路。



2. 阅读，编写并烧录以下程序，本程序将实现如下功能：右侧履带的速度将会跟随左侧履带速度，实验中用到了 PD 调节。

```
int pin1 = A0;

unsigned long duration;

#include <MsTimer2.h>

boolean flag=false;

int i=0;
```

```

int j=0;

int a=0;

int rightwheel;

int leftwheel;

void setup()
{
  pinMode(pin1, INPUT);

  Serial.begin(9600);

  MsTimer2::set(1, flash); // 500ms period

  MsTimer2::start();
}

void loop()
{
  if(!digitalRead(pin1)) i++;

  if(i>1)
  {
    i=0;

    if(j>20)

    a=j;

    j=0;

  }

  leftwheel=150;

```

```
    rightwheel=150+a-200;

    analogWrite(5,rightwheel);

    analogWrite(9,leftwheel);

}

void flash()

{

    j++;

}
```

3. 思考程序中那一段是实现速度的反馈功能？思考如何修改程序，可使小车辆侧履带速度基本一样？
4. 安装两个编码器，实现其中一条履带速度变化时（可用手调节），另一个条履带速度也发生变化。

实验 29 语音识别实验

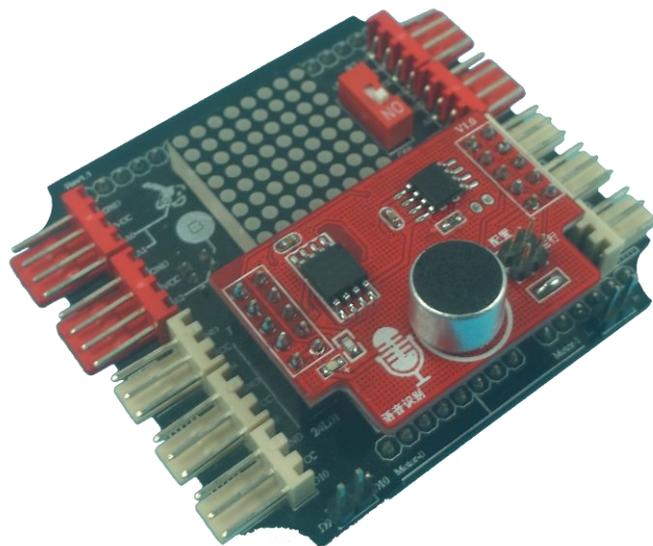
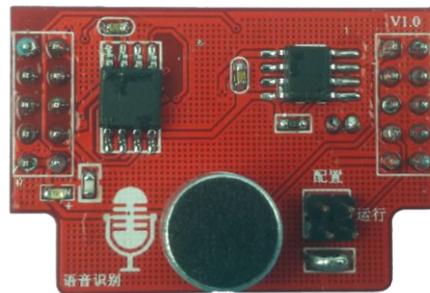
实验目的 :1. 通过语音模块实现对主控板的命令控制 ;2. 掌握语音模块的词条录入等功能 ;
3. 了解语音模块的优缺点。

实验性质 :验证型实验

实验课时 : 2 课时

参考资料 :请参考**光盘**中的“电子元件资料”, Basra 控制板, Bigfish 扩展板 ; Bigfish 连接常规传感器.ppt ; Arduino Reference。

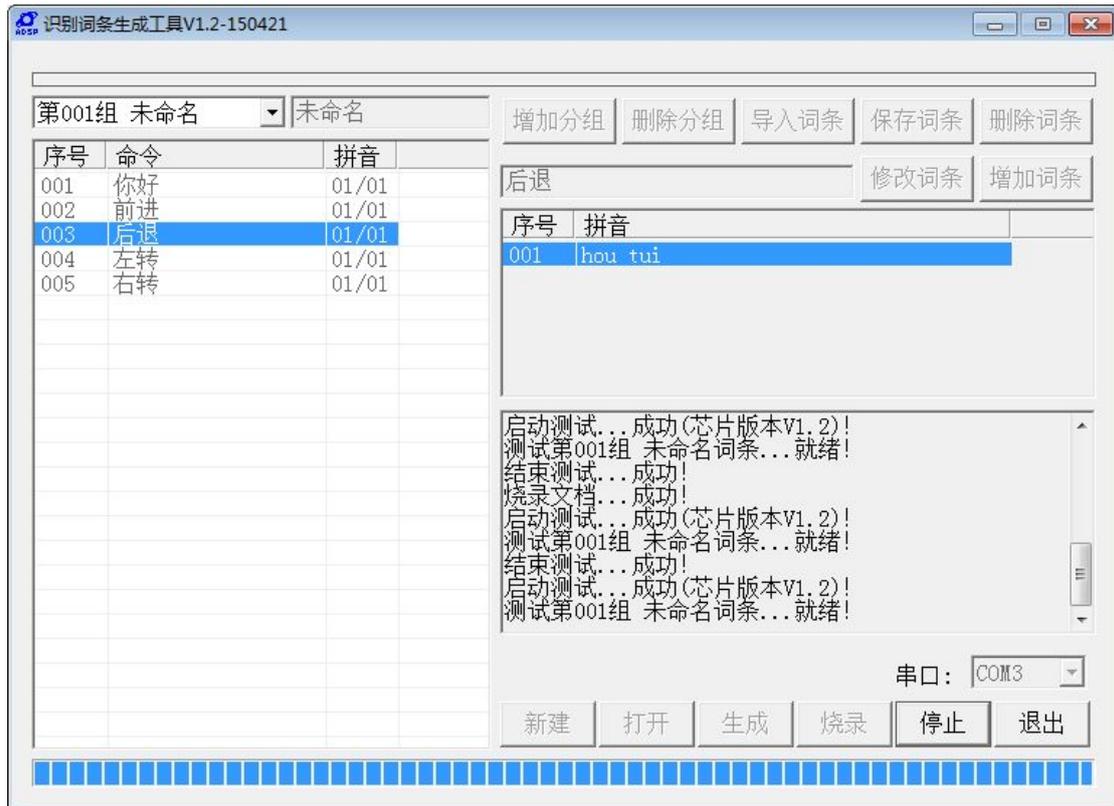
主要器材 : 语音模块×1 ; 023 号机构 ;



实验步骤 :

1. 通过软件录入特定的机组词条并烧录成功 , 上位机软件位于语音识别模块的光盘资料中。

如下图所示 :



1. 在 023 号机构上连接好电路，阅读、编写并烧录以下程序：

```

#include "HBR740.h"
#include "LedControl.h"

LedControl lc=LedControl(12,11,13,1); //点阵初始化
HBR740 hbr740; //实例化一个语音识别模块

void setup(){
    lc.shutdown(0, false);
    lc.setIntensity(0,7);
    lc.clearDisplay(0);
    hbr740.open(); //开始进行语音识别
}

void loop(){
    if(hbr740.isHeard()) //如果监测到识别语句
    {
        int value = hbr740.getSentence(); //获得识别语句的序号
    }
}

```

```
        lcOn(value); //将识别序号显示在点阵上
    }
}

void serialEvent()
{
    hbr740.lisen(); //在串口接收事件中调用语音识别的监听指令
}

void lcOn(int _value)
{
    lc.setLed(0, (_value/8), (_value%8), true);
}
}
```

3. 通过对话命令即可实现对小车的语音控制。

实验 30 双轮小车平衡实验

实验目的： 学会加速度传感器的一种应用，实现双轮小车自主平衡功能

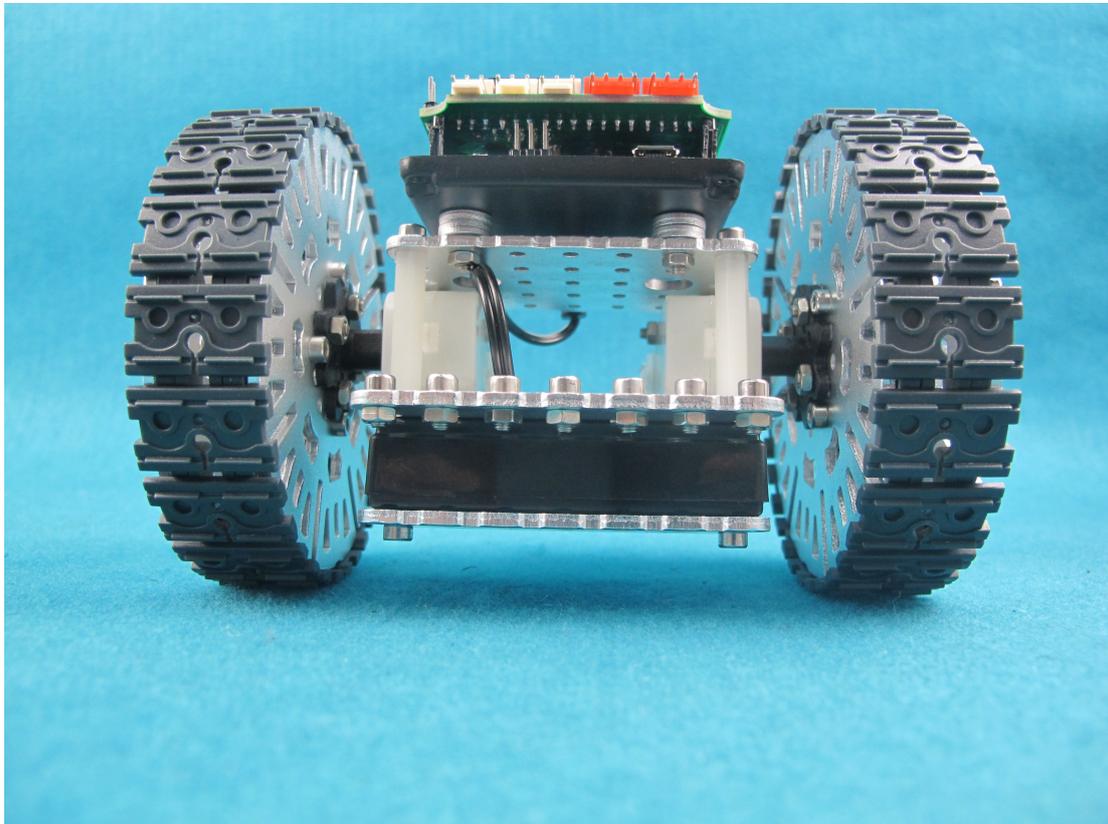
实验性质： 验证型实验

实验学时： 2

实验参考： 光盘中加速度传感器的相关文档

实验步骤：

参考光盘中的 STP 组装双轮自平衡小车，为双轮小车安装主控板、锂电池、加速度传感器等，编写代码，实现双轮小车自主平衡功能。



本实验 arduino 例程源代码如下：

```
#include <SignalFilter.h>

SignalFilter Filter;

char filtered;

int a,b,c,d;
```

```

const int analogInPin = A0; // Analog input pin that the potentiometer is
attached to

int sensorValue = 0; // value read from the pot

int outputValue = 0; // value output to the PWM (analog out)

void setup() {

    // initialize serial communications at 9600 bps:

    Serial.begin(9600);

    Filter.begin();

    Filter.setFilter('m');

    Filter.setOrder(2);

}

void loop() {

    // read the analog in value:

    sensorValue = analogRead(analogInPin);

    // map it to the range of the analog out:

    char outputValue = map(sensorValue, 0, 1023, 0, 255);

    filtered= Filter.run(outputValue);

    int a=abs(filtered);

    if((a>=91)&&(a<=100))

    {

        analogWrite(9,0);

        analogWrite(10,0);

```

```
    analogWrite(5,0);

    analogWrite(6,0);

    delay(5);}

if(a >= 101)

{

    c=a+(a-100)*5;

    d=a+20+(a-100)*5;

    analogWrite(9,0);

    analogWrite(10,c);

    analogWrite(5,0);

    analogWrite(6,d);

    delay(5);

    analogWrite(9,30);

    analogWrite(10,0);

    analogWrite(5,40);

    analogWrite(6,0);}

if(a <= 90)

{c=a+(90-a)*3+40;

    d=a+20+(90-a)*3+40;

    analogWrite(9,c);

    analogWrite(10,0);

    analogWrite(5,d);
```

```
    analogWrite(6,0);

    delay(5);

    analogWrite(9,0);

    analogWrite(10,30);

    analogWrite(5,0);

    analogWrite(6,40);

    }

    Serial.print(a);Serial.print("/");

    Serial.print(c);Serial.print("/");

    Serial.println(d);

    }
```

它将实现双轮自平衡功能，当人为的去打破小车的平衡点时，小车会自动调节至平衡点并稳定下来。

请思考：如何修改程序，可使小车实现更加良好的自平衡效果？

实验 31 蓝牙排爆机器人

实验目的：1. 使用手机蓝牙、WIFI 模块远程遥控视频小车；2. 进一步掌握串口通讯。

实验性质：设计型实验

实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板，Bigfish 扩展板；Bigfish 连接常规传感器.ppt；实验 6；实验 11；Arduino Reference。

主要器材：蓝牙模块×1；WIFI 模块×1；摄像头模块×1；023 号机构×1



实验步骤：

1. 在安卓手机上按如下步骤操作：

(1) 安装蓝牙串口助手 app

将..\“探索者”软件套装\android\蓝牙串口助手.apk 安装到手机里，或者在 google play、豌豆荚中搜索 bluetooth spp 安装，并升级，如下图所示。

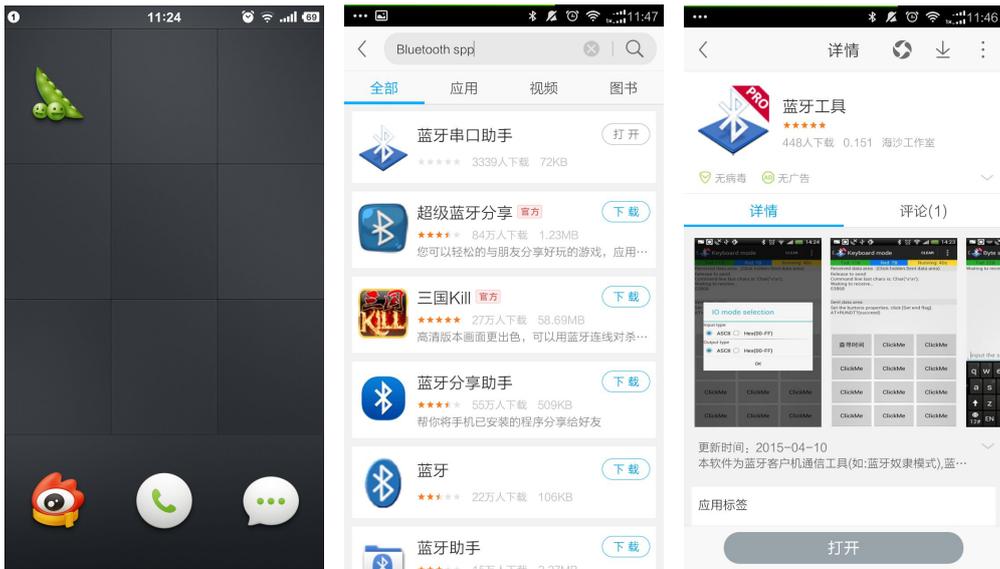


图 31.1 安装蓝牙串口助手 app，并升级

(2) 设置蓝牙串口助手

打开蓝牙串口助手 app，按如下步骤进行配置

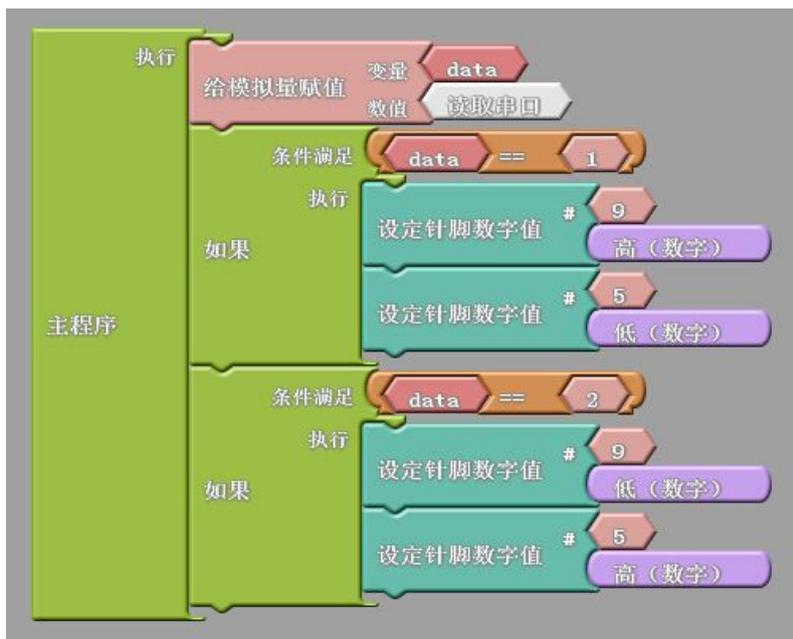


图 31.2 配置蓝牙串口助手 app



图 31.3 配置蓝牙串口助手 app

2. 将蓝牙模块连接到 Bigfish 扩展板上，并将 Bigfish 插到控制板上。接下来只要我们把串口数据和机器人要做的动作匹配起来，就可以实现用串口控制机器人运动了。用图形化程序也可以实现该功能，阅读、学习并烧录以下程序，实现用安卓手机 APP 通过串口控制 023 号小车运动。下载以下程序时，先不要堆叠蓝牙串口模块，因为会占用串口，造成下载失败。



对应的 C 语言代码为：

```

int _ABVAR_1_data = 0;

void setup()
{
  Serial.begin(9600);
  pinMode( 9 , OUTPUT);
  pinMode( 5 , OUTPUT);
}

void loop()
{
  _ABVAR_1_data = Serial.parseInt();
  if ((( _ABVAR_1_data ) == ( 1 )))
  {
    digitalWrite( 9 , HIGH );
    digitalWrite( 5 , LOW );
  }
  if ((( _ABVAR_1_data ) == ( 2 )))
  {
    digitalWrite( 9 , LOW );
    digitalWrite( 5 , HIGH );
  }
}

```

很容易看明白，串口命令“1”、“2”分别对应小车左转和右转，那么根据这个原理我们可以很容易实现用手机遥控小车前进、后退、左转、右转、停止等。同理也可以控制其他更加复杂的机构运动。

3. 自己尝试设置蓝牙按键和编程，遥控排爆机器人。
4. 如何通过 WIFI 视频组模块监控一辆 WIFI 智能小车呢？我们可以这样做：首先在 023 号机构上连接 WIFI 视频组和蓝牙模块，然后在手机端打开 WIFIROBOTS App，即可在手机端监控摄像头采集的图像；

实验 32 WiFi 视频排爆

实验目的：1. 远程监控智能小车；2. 进一步练习编程。

实验性质：验证型实验

实验课时：2 课时

参考资料：请参考**光盘**中的“电子元件资料”，Basra 控制板，Bigfish 扩展板；Bigfish 连接常规传感器.ppt；Arduino Reference.

主要器材：WiFi 模块×1；023 号机构×1；手机×1



实验步骤：

2. 组装一台 WiFi 智能小车，小车上搭载摄像头。

2. 请参考 *GL-iNET 无线路由器.word*；连接并设置无线路由器。路由器开盖教程请参考 <http://bbs.gl-inet.com/thread-14-1-1.html>，连接路由器和 Mehran 主控板的 R、T、GND 线。

3. 将 WiFiRobotV1.04_Android2.3.3.APK 下载到并安装到手机，连接 WiFi。

软件设置如下：

视频地址：<http://192.168.8.1:8083/?action=stream>

控制地址：192.168.8.1

控制端口：2001

指令设置：前进 1 后退 2 左转 3 右转 4 停止 5

4、阅读编写并烧录以下程序：

```
#define speed1 200
#define speed2 150
#define zero 0 //定义 3 个速度
unsigned long time;
void March(); //Sir,yeah Sir 前进
void Fallback(); //敌方火力太猛，后退
void Left(); //左转
void Right(); //右转
void Attention(); //稍息(就是停下来啦！)
int v = 0; // for incoming serial data

void March(){
  analogWrite(5,speed1);
  analogWrite(6,zero);
  analogWrite(9,speed1);
  analogWrite(10,zero);
}
void Fallback(){
  analogWrite(6,speed1);
  analogWrite(5,zero);
  analogWrite(10,speed1);
  analogWrite(9,zero);
}
void Left(){
  analogWrite(5,speed2);
  analogWrite(6,zero);
  analogWrite(10,zero);
  analogWrite(9,zero);
}
void Right(){
  analogWrite(6,zero);
  analogWrite(5,zero);
  analogWrite(9,speed2);
  analogWrite(10,zero); //大家小车安装不同，左转右转自己调试啦
}
void Attention(){
  analogWrite(6,zero);
  analogWrite(5,zero);
  analogWrite(9,zero);
  analogWrite(10,zero);
}
void setup() {
  Serial.begin(9600);
  pinMode(5,OUTPUT);
  pinMode(6,OUTPUT);
  pinMode(9,OUTPUT);
  pinMode(10,OUTPUT); // opens serial port, sets data rate to 9600 bps
}
void loop() {
  // send data only when you receive data:
  if (Serial.available() > 0) {
    // read the incoming byte:
```

```
        v = Serial.read();
        // say what you got:
        Serial.print("I received: ");
        Serial.println(v, DEC);           //接收串口数据
    }
    if(v==1)
    March();
    if(v==2)
    Fallback();
    if(v==3);
    Left();
    if(v==4);
    Right();
    if(v==5)
    Attention();
}
```

5. 打开手机 WiFi 小车 APP，进入开始选项，即可实现智能小 车远程视频监控。



机器时代（北京）科技有限公司

ROBOTTIME BEIJING TECHNOLOGY. LTD

电话：010-60705356

Email：web@robottime.cn

<http://www.robottime.cn>