第四章 乐高传感器

本章所包含的内容：

- 触动传感器
- 光电传感器
- 角度传感器
- 温度传感器
- 传感器的使用方法与技巧
- 其他传感器

4.1 简介

马达通过齿轮和滑轮传动，可以让你搭建的机器人动起来，他们就如同是移动机器人腿和手臂的肌肉。同时，你还可以使用传感器来装备你的机器人，它们就如同是机器人的眼睛、耳朵和手指。

机器人套装中包含两种传感器：触动传感器（两种）和光电传感器。在本章中，我们主要是描述它们的特性，对于其它的传感器你可以单独购买，如：角度传感器和温度传感器。每一个设备都有其特定的作用，你将会因为它们的功能强大和所能涉及的范围之广而感到惊讶。当然也包括这种情况,可以用一种传感器仿效另一种传感器，以用来代替不能使用的传感器。利用 RCX 上的红外光电，使用一些小技巧，你可以把把你的光电传感器变成一个雷达。

在阅读本章的过程中,我们希望你能把机器人套装放在身边,这样你可以跟随我们的例子亲自动手去做。为了保持其完整性，我们还会讲一些机器人套装的扩展套装和技术套装的内容。若你现在还没有这些也不要担心，这不会影响到你搭建体积较大的机器人。

4.2 触动传感器

触动传感器（图 4.1）是乐高传感器大家庭中最简单、最直观的一种。它的工作方式非常像是你家门铃上的按钮：当它被按下时，电路接通，电流就会通过，RCX 就能够检测到这个数据流，你的程序就会读取触动传感器的当前状态：开或者关。

图 4.1 触动传感器

如果你已经开始使用机器人套装，阅读了 Constructopedia，并搭建了一些模型，你可能对传感器的一般用途比较熟悉，如缓冲器。缓冲器是与周围环境相互作用的一种简单方式，当你的机器人遇到障碍物时，可以用它们来进行检测，并由此而改变运动状态。

典型的缓冲器是一个重量较轻的可移动装置，事实上，当它碰到障碍物时会把冲击力传递给触动传感器并使之关闭。你也可以发明出很多种缓冲器，但它们的外形应该能够反映机器人的外形，而且还能反映出环境中障碍物的外形。如图 4.2 中所示一个非常简单的缓冲器，可以很容易发现墙壁，假如房间里有像椅子一样等复杂障碍物，它的效果就不好了。在这种情况下，我们建议你通过实验来进行。为机器人设计一个缓冲器，在房间的周围离地板适当高度的地方移动它，检查它是否能够发现所有可能的碰撞点。如果你的缓冲结构较大，当它用最佳部位撞击到障碍物并按下触动传感器时，不要以为这就是正确的。图例 4.2 是一个不太好的缓冲器，因为当碰撞发生时，它几乎不能用横轴的边缘来关闭触动传感器，说它是一个不好的缓冲器是因为它把整个碰撞产生的力直线传输给了传感器，也就是说，在机器人身上安装一个非常稳固的支架对传感器的安装是非常有必要的。

图 4.2 简单的缓冲器

根据经验，应该尝试不同的碰撞来看看缓冲器在各个位置是否能很好的工作。你可以编写一个无限循环的小程序，当传感器被按下时，发出一声蜂鸣声用来测试你的缓冲器。

谈起缓冲器，人们往往会想到当遇到障碍物时开关会被按下。这样说似乎有点绝对，在发生碰撞的时候同样可以松开开关。看一下图 4.3，橡胶皮筋可以使积木轻压着传感器，当缓冲器的前面部分接触到物体时，开关就会被释放。

图 4.3　平常压下的缓冲器

实际上，向你推荐这种缓冲器有这样几个重要的原因：

- 冲击力不可能直接传递给传感器，传感器与标准的乐高积木相比更容易受损坏，因此应避免不必要的撞击。

- 橡皮圈可以吸收撞击力，这对你的传感器和机器人来说都会起到保护作用，当你的机器人速度非常快，或者重量大，或者反应比较慢，或者具备其中的一个因素时，这种保护作用显得尤其重要。

缓冲器是一个非常重要的装置，而且触动传感器的应用也非常广泛。当你想告之 RCX 一个事件发生时，你可以使用按钮一样人为的把它按下去。你能想象出类似的情况吗？事实上，有很多。比方说，你可以按下铵钮告诉 RCX”现在读取光电传感器的值”，从而进行读取校准（我们将在以后的部分进行讨论）。

另一个常用的作用就是把触动传感器作为一个位置控制器来用。如图 4.4，机器人向前看时（图 4.4b）就关闭了头部（图 4.4a）的触动传感器。通过编程可以在水平面上实时的控制头部的旋转（向左或向右），当传感器被按下时，机器人的头部就能转回到正确的位置，值的注意的是，我们在这个例子中用到的凸齿轮在与触动传感器相配合时是非常的有用，你可以让轴通过三个十字孔个中的一个来选择合适的距离去关闭触动传感器。

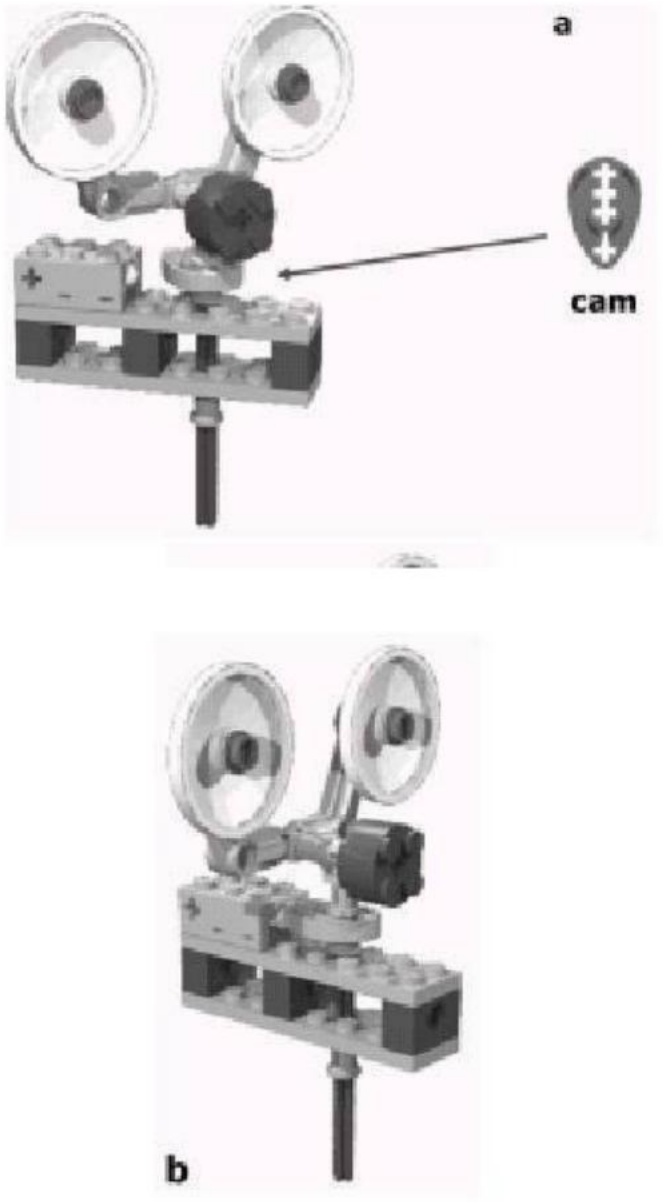

图 4.4　用触动传感器定位

在本书的第三部分我们还会介绍位置控制的其它一些应用。事实上，在搭建你的机器人之前需要你去研究一些不同的方法。

我们再举几个事例来说明传感器的应用。假如你想搭建一个电梯。你希望电梯可以在任何一层都能停下。首先你会想到在每一层安放一个触动传感器，当按下其中一个时，电梯室会在那一层停下。这是一个好方法，但这里有一个小问题：你仅有两个触动传感器，对你来说，一个电梯仅有二层不是非常实际，你可以再买一个触动传感器，但这只能再加高一层,并没有解决实质性的问题。此时，RCX 的三个输入端口已经全部被占用。突然，你会想到一个办法：为什么不把传感器放在电梯厢上而不是外面呢？在电梯厢上固定一个传感器，这样只需一个传感器就可以加高更多的楼层了。从最初的方案到现在更好的方案，两个系统是完全相

同的吗？答案是否定的。首先，你需要决定厢的绝对位置，当它在第二层时，你仅是可以知道它的相对位置。那么，你需要一个初始点，从起始点开始计算就可以推断出厢的位置.或者程序运行时，需要厢体在一个特定的位置，或者用第二个传感器来探测一个特定的楼层。例如，在最底层放置一个传感器，因此程序一开始就降下电梯到最底层。那样就可以计算出厢体的绝对位置了。

现在，电梯就能够准确的升降了。但你还有最后一个问题需要解决：如何告诉你的电梯它应该去哪一层呢？在每一楼层放置一个传感器去提示电梯是不切合实际的。在 RCX 上只剩下一个输入端口了。你要用这一个触动传感器来做什么呢？你还可以采用以前的方法吗？可以，你可以计算一个触动传感器被按下的次数。比如说，被按下三次表示是三楼，依次类推。现在你就可以去搭建你的电梯了。

方法与技巧

计算按下次数

下面这个例子是用伪代码来编写的，一个代码并不与实际编写语言相对应，而是界于程序语言和机器语言之间。使用伪代码编程在专业程序员中是一种非常普遍的做法。

计算触动传感器被按下多少次需要一些小窃门。假如你写了一些简单的代码，如下：

```
Counter =0

repeat

     if   Sensor1 is on then

            Counter   =   Counter +1

end if

end     repeat
```

当你保持触动传感器被按下很短的间隔内，你的程序代码就会在你的 RCX 上飞快的执行。然而，在记录下一个新的按下之前，需要松开触动传感器：

```
Counter   = 0

repeat

       if   Sensor1   is   on then
```

```
                    Counter =Counter+1

                    wait until   Sensor1   is off

end if

end repeat
```

现在，你编写的代码正确地记录了从断开到闭合的变化。在你的代码中，有一个重要的特点需要介绍一下：当它在一段时间内接收不到触动信息时，你希望你的计数程序停止。为了实现这一点，你需要使用一个计时器用来测量上一次按下时间与最后一次时间的间隔：

```
Counter=0

interval = <a proper   valve>

reset Timer

repeat

       if   Sensor1   is on then

              Counter = Counter + 1

wait until Sensor1   is off

                     or until Timer if greater then Interval

           reset Timer

end if

until Timer is greater then   Interval
```

假如你的时间间隔是两秒。当计数程序开始时，计时器和计数器首先复位为 0，然后开始检测传感器的状态。如果两秒内没有按下触动传感器，它仍将保持复位状态，如果有触动传感器被按下，此动作将被记下并等待使用者释放按钮，计时器复位为零， 在程序停止运行之前，使用者可以在两秒内第二次按下触动传感器。

4.3 光电传感器

用”看”来形容光电传感器的功能有点夸大其辞。实际上它只是用来检测光并测量其强度。尽管受到限制，但其应用范围仍比较广。

图 4.5　光电传感器

光电传感器和触动传感器的最大区别是，后者返回的是一些数值而不是单纯的开/关状态。你所读到的数值由光电传感器在那个时候所检测到的光强所决定。这些数值以 0 至 100 的百分数的形式返回，光值越大，百分数就越大。你可以用光电传感器来做什么呢？你可以用它搭建一个由光电传感器所驱动的机器人，我们称之为光的追随者，它可以检测周围的环境，找到一个强光源（或者是最亮的）并朝着它前进。在一间足够暗的房间里进行，以免产生干扰，你也可以用手电筒来控制你的机器人。

检测外部光源的功能是非常有趣的，但是或许你不能用它来做最令人惊奇的事情。我们介绍一下光电传感器的另一个用途：它不但可以用来检测光强，而且还可以自身发光。提供稳定光源的是一个红色的发光二极管，因此你可以用来测量反射光并传给传感器。

当你用来测量反射光时，你必须去避免一些来自其它光源的干扰。需要注意的是光电传感器对 IR 所发出的光也非常敏感，像典型的远距离控制器发出的红外光，如摄像机；或者是乐高红外发射器。

设计与计划

读取周围的光值

乐高光电传感器并不适合于测量外部光源来说，因为其灵敏度比较弱。红色发光二极管所所发出的光太靠近检光器以致于过多的影响了光值的读取。如果你想测外部光源，你应该考虑尽可能的去减少红色发光二极管的影响。一个简单的办法就是在光电传感器的前部放一个 1×2 的单孔积木块，.更多的行之有效的办法需要你对光电传感器有些细微的改动。在 Ralph Hempel 的网站中，他提到了如何对光电传感器作修改，既不是永久的改变也不会损害你的光电传感器。（见附录 A）

光在表面的反射率取决于许多因素，主要是表面的颜色，质地和它距光源的远近。黑色物体的反射能力要弱于白色物体；黑色光滑表面的反射能力要强于黑色不光滑表面。另外，距离光电传感器越远，光电传感器所检测到的反射光就会越少。

这些因素都是相互依赖的，通过光感读取的值，并不能说明是由哪个因素引起的。但你可以

保持其它因素不便，而让一个变化，这样就可以通过读取的数值来推断出环境的一些变化。例如，如果你的光电传感器经常对着同一个物体，或者相同质地和颜色的物体，你能够用它去测出它的相对距离。另一方面，你可以把不同的物体放置在光电传感器的前面，在恒定的距离内分辨出它们的颜色。

4.3.1 测量反射光值

为了举例说明测量反射光的原理，我们来做一个实验。拿一个 RCX 并打开其电源，在任意一个输入端口连接一个光电传感器，在你的程序中正确的设置其端口（红色发光二极管应该发光）。准备一间光线较暗的房间，RCX 有个控制模式，可以实时查看光感读取的值。按 RCX 上的 VIEW 键，当小箭头正确显示在传感器所接的输入端口位置。在显示屏上会显示出读取的光线值。接下去你把光感放到桌上，在桌上以一定间隔（0.5cm,1cm,1.5cm）并排放置不同颜色的积木块,保证积木之间的间距相等。查看数据，得到的是不同颜色的积木反射的光值是不同的。

再进行第二个实验：将白色的积木快慢慢的移向光感，然后再慢慢的移开，观察显示屏上的数据，可以发现当光感与积木间的距离加大时数值将减小。我们的目的就是为了证明光感是不能同时判断出距离和颜色的。我们重点强调在你使用光感时尽量避免外界光线的干扰。

方法与技巧

理解原始值的概念

了解原始值是很关键的，对于熟练的使用机器人套装并不是必须的。但从另一个方面讲，它可以帮助你理解传感器是如何工作的。

RCX 把来自传感器（不管是什么类型）的电信号全部转变成范围在 0 至 1023 之间的数字信号，我们称之为原始数值。在程序中，你在一个端口上设置了一个特定的传感器，RCX 会自动设定该传感器的原始数值范围，例如，从触动传感器读取的数值范围是 1 或者 0，代表开或者关，当从温度传感器读取时就会转变成摄氏温度或华氏温度。同样地，光电传感器读取时就会通过下面的方程式转换成一个百分数：

百分数=146-原始数值/7

为什么我们需要知道这种转换呢？对于大多数应用程序来说，通过 RCX 返回的光感值的百分数形式更加有效，但也有这种情况，你需要所有光感变化值而百分数形式却不能体现出光感值的一些变化。我们用一个例子来做一下说明。假设会有两种不同的情形，光电传感器读取 707 和 713 两个不同的数值。把这两个数值转换成百分数，因为 RCX 只使用整数,需要将除的结果进行圆整。

146-(707/7)=146-101=45

146-(713/7)=146-101=45

在第二个方程式中数值 101 实际上是 101.857…，去掉小数部分为 101.就看部不出两个值是不同的。我们知道在大多情况下数值的小数部分是不重要的。但也会有其它情况需要用到这样一个微小的变化的数据

如果你用 LEGO 图形化的编程环境为 RCX 编写程序，你必须接受它的刻度值，否则无法处理原始数值。如果你选择其它的编程语言，则可以直接处理没有处理过的原始数值，在必须的时候，利用其优势，可能会有更好的解决方案。

识别不同的颜色是光电传感器一个非常普通的应用。我们曾说过，光电传感器实际上并不是用来识别颜色的，而是用来读取反射光值。因此，它很难把黑色和蓝色的积木块区别开。但目前来说，我们仍说它能识别颜色，在读完之后您会明白真正的意思是什么。

4.3.2 沿线走

目前，光电传感器最普遍的使用方法就是用它搭建一个沿线走的机器人。

这个项目的设置是很简单的，这也是之所以流行的一个原因。尽管其外观简单，这项工作仍需要引起足够的重视，并需要你仔细设计和认真编程。我们将在第二部分详细的讨论这个主题的细节内容。当光电传感器在轻质地面上读取一条黑线时，你要注意有什么事情发生。

当把光电传感器放在地板上时，假设说读取的数值为百分之七十，黑线为百分之三十。如果你想让机器人缓慢的从地板移动到黑线或者是有污点的地方.你会注意到，数值不是从一个值突然跳到另一个值,而是会出现一系列的中间值。原因是光电传感器不是读取一个点，而是光电传感器前部的一个小区域。所以当光电传感器穿过线的边缘时，它所读取的是地板和黑线的边界值并返回一个中间值。

这个功能有用吗？当然，有时有用，有时没有用。尤其当我们涉及到沿线走时，它是有用的。实际上，你可以（或者说应该）编写一段程序让你的机器人沿着边界走而不是实际的黑线。这样当机器人需要改变它的行进路线时，它知道往哪个方向转身：如果所读取的值太”暗”，它应该向亮的区域前进，或有污点的区域。

技巧与提示

校准读取值

有时，你并不能预先知道光电传感器实际上所读取的数值是多少。假如你要参加一个沿线走的比赛：你并不能确定你的传感器所返回的地板和黑线的数值。在这种情况下，一般的习惯是，在你的程序中不写入预期的常数值是比较好的。但可以让你的机器人通过一个简单的测量程序来读取这些数值。继续我们沿线走的例子,你可以专设一个空的输入端口用来接入一个触动传感器，当你把机器人放在地板上时手动按下触动传感器，然后再放在黑线上，因此它就可以保

存下读取到的最大值和最小值。或者你可以编写一个小的检测程序，以取消那些限制。

当你需要控制一个更复杂的区域时,举例来说，区域包括三种不同的颜色， 想象一个台面被划为白色，黑色和灰色三个不同颜色的区域。在白色和黑色之间的边界上你如何能区别出灰色区域？这时你不能只做一个简单的读取，你必须深入考虑其它的因素，像预先读取,或者你可以使你的机器人在一个地方收集更多的数据并推断它所在的位置。要处理这样的情况，对软件的要求就会变得更加复杂。

光电传感器如同一个万能器，它有很多种使用方法。你可以在光电传感器的前部放置一个彩色的可以移动的乐高梁来搭建一个对称形状的装置.图 4.6 就是一个这样的例子。当你推或者拉梁的上部时，光电传感器就会读取不同的光值。

图 4.6 用光电传感器作一个模拟控制

光电传感器与灯相结合（不包括在头脑风暴套装中）可做成一个光电管（图 4.7）；当有物体挡在光电传感器和灯之间时，机器人就会察觉。值得注意的是，我们在光电传感器的前面放置了一个 1×2 的单孔梁，以减少来自周围光线的干扰。

图 4.7　光电管

4.3.3 接近探测

你可以用光电传感器做成一个雷达探测器用于检测即将碰到的障碍物。这被称之为接近探测。这项技术所基于的特性我们已经讨论并探究过了，就是光电传感器可以根据反射光线来

测定相对距离。假如你的机器人要直线前进，用一个光电传感器为它在前面引路。假如你的机器人要在一个暗室里移动，除了光电传感器上的红色发光二极管之外没有任何的光源。在向前移动的过程中，机器人连续不断的读取传感器所检测到的光值。如果读取值趋向于迅速增大，就可以推断出机器人正向着一个物体前进。但不能推断出障碍物的种类及与障碍物之间的距离，如果房间内没有物体在移动，你确信机器人正在接近障碍物。现在我们有了一套系统可以躲避障碍物而不是局限于碰撞以后再检测它们。

注意：

RCX 内部的 IR LED 发射的是不可见光，光电传感器的红色二极管发射的是可见光。

遗憾地是，当房间内有光源时，这项技术工作就有问题了，原因是你的程序不能区别自身反射回的红色光还外界环境光线。你需要在机器人身上有一个更明显的独立光源提供更高的参考。

令人欣慰的是，正好有一个！RCX 内部有一个 IR LED 可以发射信息给红外发射仪或是另一个 RCX。用 RCX 内部的 IR LED 以比特的编码形式发送信息可以被红外发射器所接收到。关于信息的内容我们并不关心;我们需要的仅仅是光。尽管红外光对于肉眼来说是不可见的，却与可见光具有相同的性质，LEGO 光电传感器对此却非常敏感。

所以，现在你的程序有了使用接近探测的所有条件。发送一个 IR 信息并立即读取光电传感器的值.你最好把读取的数值进行一下平均处理，这样可以把外部光源所导致的影响降至最低（我们将会第 12 节讨论这个窍门）。如果你注意到在随后的二组值中有显著的增加，举例来说，百分之十，说明你的机器人很有可能正朝着障碍物前进。

4.4 角度传感器

我们将要研究的第三个乐高传感器是角度传感器（图 4.8）。遗可是机器人套装中没有包含该部件，它的多功能性仅次于光电传感器。在 3801 Ultimate Accessory 套装里面包含一个角度传感器，还有一个触动传感器，一个灯,遥控器 和少量的其它附件。

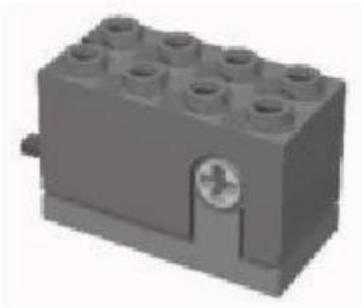

图 4.8　角度传感器

方法与技巧

角度传感器是如何工作的呢？

因为角度传感器有四种不同的状态，所以会返回四种不同的值。我们称之为 A，B，C 和 D。对于每一次完整的旋转，它经过了四种状态各四次—这也就是我们为什么要用十六来计数的原因。如果角度传感器是顺时针旋转，它会读序列 ABCD…，如果是逆时针旋转，读取的结果会是 ADCBA…，RCX 会时刻检测传感器，当 RCX 检测到状态发生变化时，它不但可以推断出角度传感器已经转动，而且还可以知道所旋转的方向。举例来说，从 A 转变到 B，或从 D 转变到 A，计数器将增加一个单位，然而，从 D 到 C，或者是从 A 到 D，计数器将减少一个单位。

角度传感器，顾名思义，是用来检测角度的。它的身体中有一个孔，可以配合乐高的轴。当连结到 RCX 上时，轴每转过 1/16 圈，角度传感器就会计数一次。往一个方向转动时，计数增加，转动方向改变时，计数减少。计数与角度传感器的初始位置有关。当初始化角度传感器时，它的计数值被设置为 0，如果需要，你可以用编程把它重新复位。

通过计算旋转的角度，你可以很容易的测出位置和速度。当在机器人身上连接上轮子（或通过齿轮传动来移动机器人）时，可以依据旋转的角度和轮子圆周数来推断机器人移动的距离。然后就可以把距离转换成速度，你也可以用它除以所用时间。实际上，计算距离的基本方程式为：

距离=速度×时间

由此可以得到：

速度=距离/时间

如果把角度传感器连接到马达和轮子之间的任何一根传动轴上，必须将正确的传动比算入所读的数据。举一个有关计算的例子。在你的机器人身上，马达以 3:1 的传动比与主轮连接。角度传感器直接连接在马达上。所以它与主动轮的传动比也是 3:1。也就是说，角度传感器转三周，主动轮转一周。角度传感器每旋转一周计 16 个单位，所以 16*3=48 个增量相当于主动轮旋转一周。现在，我们需要知道齿轮的圆周来计算行进距离。幸运地是，每一个 LEGO 齿轮的轮胎上面都会标有自身的直径。我们选择了体积最大的有轴的轮子，直径是 81.6CM(乐高使用的是公制单位)，因此它的周长是 $81.6\times\pi=81.6\times3.14\approx256.22$CM。现在已知量都有了：齿轮的运行距离由 48 除角度所记录的增量然后再乘以 256。我们总结一下。称 R 为角度传感器的分辨率（每旋转一周计数值），G 是角度传感器和齿轮之间的传动比率。我们定义 I 为轮子旋转一周角度传感器的增量。即：

$I=G\times R$

在例子中，G 为 3，对于乐高角度传感器来说，R 一直为 16.因此，我们可以得到：

I=3×16=48

每旋转一次，齿轮所经过的距离正是它的周长 C，应用这个方程式，利用其直径，你可以得出这个结论。

C=D×π

在我们的例子中：

C=81.6×3.14=256.22

最后一步是将传感器所记录的数据-S 转换成轮子运动的距离-T，使用下面等式：

T=S×C/I

如果光电传感器读取的数值为 296，你可以计算出相应的距离：

T=296×256.22/48=1580　　距离（T）的单位与轮子直径单位是相同的.

实际上，在程序不仅仅会用到乘法和除法的数学运算，还有更多的需要多留心（有关内容我们将在第 12 章进行进一步的讨论）。

使用角度传感器来控制你的轮子可以间接的发现障碍物。原理非常简单：如果马达运转，而齿轮不转，说明你的机器已经被障碍物给挡住了。此技术使用起来非常简单，而且非常有效；唯一要求就是运动的轮子不能在地板上打滑（或者说打滑次数太多），否则你将无法检测到障碍物。如果是一个空转的齿轮连接到马达上就可以避免这个问题，这个轮子不是由马达驱动而是通过装置的运动带动它:在驱动轮旋转的过程中，如果惰轮停止了，说明你碰到障碍物了。

在许多情况下角度传感器是非常有用的：控制手臂，头部和其它可移动部位的位置。值的注意的是，当运行速度太慢或太快时，RCX 在精确的检测和计数方面会受到影响。事实上，问题并不是出在 RCX 身上，而是它的操作系统，如果速度超出了其指定范围，RCX 就会丢失一些数据。Steve Baker 用实验证明过，转速在每分钟 50 到 300 转之间是一个比较合适的范围，在此之内不会有数据丢失的问题。然而，在低于 12rpm 或超过 1400rm 的范围内，就会有部分数据出现丢失的问题。而在 12rpm 至 50rpm 或者 300rpm 至 1400rpm 的范围内时，RCX 也偶会出现数据丢失的问题。

这仅仅是一个小小的问题，你可以上下调整传感器来使其处在合适的范围内。