第1章　视　错　觉

1.1　直线变弯

作品描述

该作品用于验证一种经典的视错觉——黑林错觉，作品效果见图 1-1-1。19 世纪初，德国心理学家艾沃德·黑林发现，平行的直线在放射线的干扰下，会使人对线条和形状的感知产生错误，认为直线发生了弯曲。

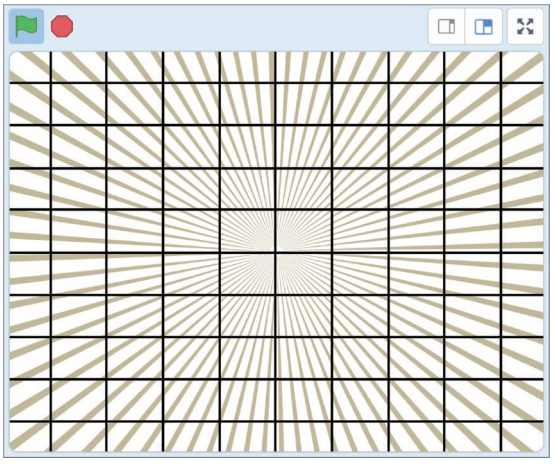

图 1-1-1　作品效果图

创作思路

绘制由水平和垂直的两组平行直线构成的网格，并控制网格在舞台上移动，使“直线变弯”的错觉更容易被观察。为简化编程，利用克隆技术生成水平和垂直的两组直线克隆体，并使用滑行运动积木使这些直线克隆体在舞台上平滑移动。

编程实现

先观看资源包中的作品演示视频 1-1.mp4，再打开模板文件 1-1.sb3 进行项目创作。

（1）导入有放射线的背景图。

从 Scratch 背景库中找到名为 Rays 的图片，将其导入项目中作为舞台的背景。Rays 图片包含一组由中心向四周发散的放射线，可以直接在本项目中使用。

（2）创建“横线”角色并编写代码。

首先创建一个空角色，并将角色名称修改为“横线”；然后使用绘图编辑器工具栏中的线段工具在画布上画出一条与舞台等宽的直线段，并将其大小调整为 4，轮廓颜色设为黑色；接着切换到“横线”角色的代码区，编写如图 1-1-2 所示的代码。

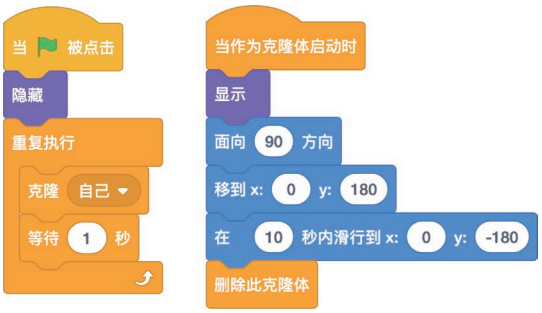

图 1-1-2 “横线”角色的代码

当项目运行后，“横线”角色负责不断地创建水平方向的直线克隆体，并使其从舞台的顶部边缘在 10 秒内平滑移动到舞台底部边缘。

（3）创建“竖线”角色并编写代码。

在角色列表中将“横线”角色复制一份，并将新角色的名称修改为“竖线”。然后，切换到“竖线”角色的代码区，编写如图 1-1-3 所示的代码。

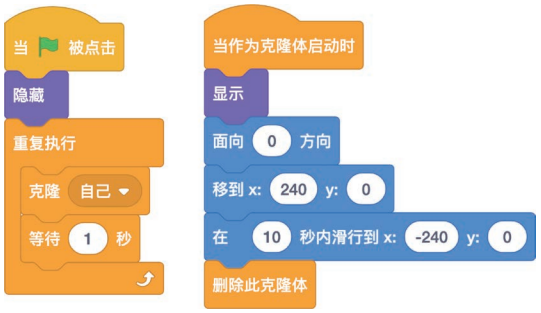

图 1-1-3 “竖线”角色的代码

当项目运行后，“竖线”角色负责不断地创建垂直方向的直线克隆体，并使其从舞台的右侧边缘在 10 秒内平滑移动到舞台的左侧边缘。

当两个角色的代码同时运行后，将在舞台上产生一个平滑移动的网格。由于受到背景图中放射线的干扰，可以观察到直线变弯的假象。并且，在舞台中部放射线密集的地方，直线弯曲的幅度更大。

1.2　闪现的暗点

作品描述

该作品用于验证一种栅格交叉点上明暗飘忽、闪来闪去的视错觉——栅格错觉，作品效果见图 1-2-1。最初的栅格错觉是由卢迪马尔·赫尔曼在 1870 年发现的，栅格图案的构成相当简单，由黑色方块整齐排列，中间空出了垂直相交的白色条纹。在观察这种栅格图案时，观看者的余光会看到各个白色条纹的交叉处存在着暗点，而只要视线中心转移到那里，暗点就会消失。这种错觉的强度取决于图像中黑白对比的程度，以及观察者的视觉感知。如果把黑色方块换成其他颜色，暗点看起来依然存在。

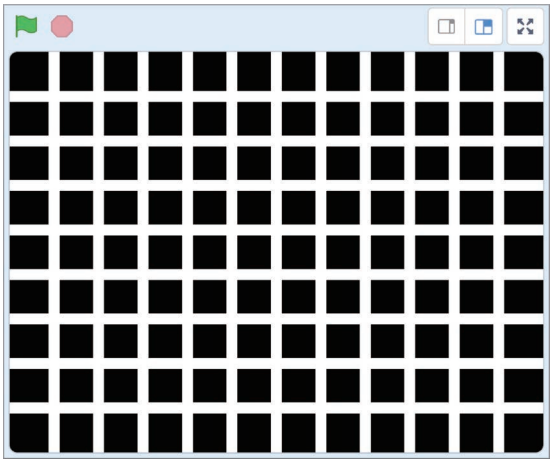

图 1-2-1　作品效果图

创作思路

绘制由水平和垂直的两组白色线条构成的网格，舞台背景使用黑色填充，也可以使用其他颜色。

编程实现

先观看资源包中的作品演示视频 1-2.mp4，再打开模板文件 1-2.sb3 进行项目创作。

(1) 编写主程序的代码。

在主程序中，将画笔的颜色设置为白色，将画笔的粗细设为 10，然后依次调用“画横线”和“画竖线”这两个过程绘制出白色线条构成的网格图案。

(2) 编写“画横线”过程和“画竖线”过程的代码(见图 1-2-2)。

在“画横线”过程中，通过一个条件型循环结构绘制出垂直方向间隔为 40 个单位的一组白色横线。由变量 y 控制横线出现的 y 坐标，y 坐标由 -140 变化到 140。横线的长度为 480 个单位，即起点的 x 坐标为 -240，终点的 x 坐标为 240。

在“画竖线”过程中，通过一个条件型循环结构绘制出水平方向间隔为 40 个单位的一组

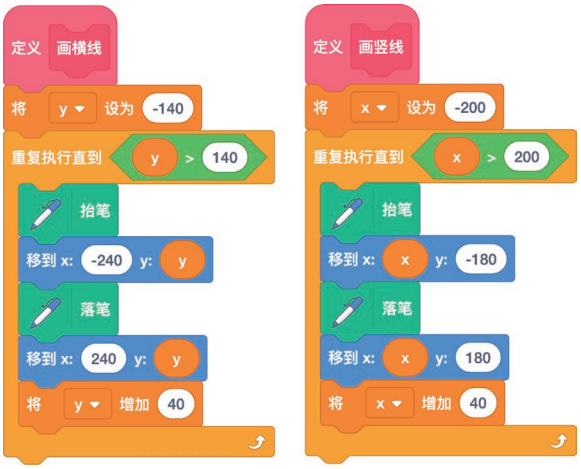

图 1-2-2 “画横线”和“画竖线”过程的代码

白色竖线。由变量 x 控制竖线出现的 x 坐标，x 坐标由 -200 变化到 200。竖线的长度为 360 个单位，即起点的 y 坐标为 -180，终点的 y 坐标为 180。

当项目运行后，可以切换到全屏模式，这样更便于观察呈现出的视错觉效果。

1.3 灰色变蓝色

作品描述

该作品用于验证一种灰色变成蓝色的视错觉，作品效果见图 1-3-1。当眼睛注视右边的黑色圆点时，通过余光可以看到左边橘色矩形区域中的灰色方块会变成蓝色的；但当视线离开黑色圆点时，它又恢复为灰色的。当灰色方块上下不断移动时，这种颜色错觉表现得更为强烈。

图 1-3-1 作品效果图

创作思路

在舞台左侧放置一个“灰色方块”角色，并控制其在垂直方向上反复进行平滑移动。

编程实现

先观看资源包中的作品演示视频 1-3.mp4，再打开模板文件 1-3.sb3 进行项目创作。

(1) 角色的准备。

利用绘图编辑器绘制出 3 个角色的造型。深灰色(颜色 0、饱和度 0、亮度 50)的长方块放置在舞台的左侧，并通过代码控制它进行上下运动，橘色的方块铺满舞台的左半部分，黑色的圆形放置在舞台的右下角。另外，将舞台背景填充为浅灰色。

(2) 编写“灰色方块”角色的控制代码(见图 1-3-2)。

图 1-3-2　控制灰色方块运动的代码

在这个作品中只需要编写“灰色方块”角色的代码。使用“重复执行”积木控制“灰色方块”角色上下反复运动。起始位置为(－130，100)，结束位置为(－130，－100)，使用“在……秒内滑行到……”积木实现角色的平滑移动。

当项目运行后，可以切换到全屏模式，这样更便于观察呈现出的视错觉效果。

1.4　变大又变小

作品描述

该作品用于验证一种对实际大小感知上的视错觉——艾宾浩斯错觉，作品效果见图 1-4-1。德国心理学家赫尔曼·艾宾浩斯发现，在中心圆具有相同半径时，人会觉得被小

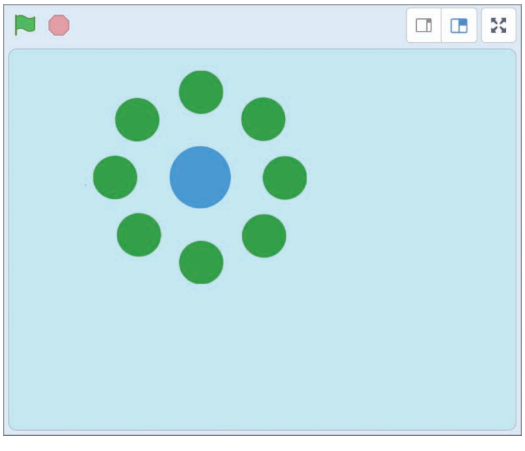

图 1-4-1　作品效果图

圆包围的中心圆比被大圆包围的中心圆更大。该作品运行后，会在运动中不断地改变包围在蓝色球周围的一组绿色球的大小，而蓝色球的大小保持不变，但却能让人观察到蓝色球忽大忽小的错觉现象。

创作思路

该作品采用动画进行展示，控制“蓝色球”角色和“绿色球”角色在舞台上进行平滑运动或改变大小，从而让观察者直观地感受艾宾浩斯错觉。

编程实现

先观看资源包中的作品演示视频 1-4.mp4，再打开模板文件 1-4.sb3 进行项目创作。

(1) 编写“蓝色球”角色的代码(见图 1-4-2)。

使用“重复执行”积木控制“蓝色球”角色在(−100,100)到(0,0)两点之间反复运动，使用“在……秒内滑行到……”积木实现角色的平滑移动。

(2) 编写“绿色球”角色的代码。

如图 1-4-3 所示，使用“重复执行”积木控制“绿色球”角色在(−100,100)到(0,0)两点之间反复运动，使其始终包围在蓝色球周围。在“绿色球”角色的运动过程中，通过“广播‘放大’”和“广播‘缩小’”这两个消息不断地改变角色大小。

图 1-4-2　控制“蓝色球”角色运动的代码

图 1-4-3　控制“绿色球”角色运动的代码

如图 1-4-4 所示，“绿色球”角色在接收到“放大”或“缩小”的消息后，分别利用计时器积木控制角色在 1 秒之内不断地改变角色的大小。

当项目运行后，可以切换到全屏模式，这样更便于观察呈现出的视错觉效果。

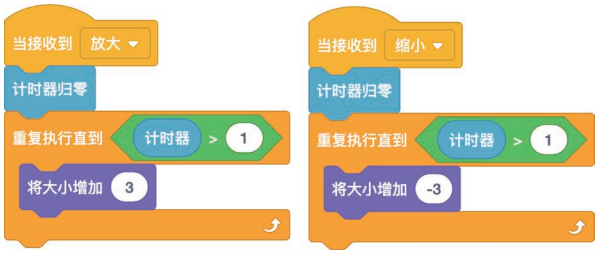

图 1-4-4　控制“绿色球”角色放大和缩小的代码

1.5　无中生有的绿点

作品描述

该作品用于验证经典的追逐丁香视错觉，作品效果见图 1-5-1。英国视觉专家杰里米·辛顿(Jeremy Hinton)在 2005 年创作了这个视错觉动态图像，它由 12 帧静止画面组成，每两帧之间间隔约为 0.1 秒，每一帧画面中心黑色十字的周围都环绕着 11 个紫色的圆点。周围圆点总数应该为 12 个，但是每一帧都缺失 1 个圆点，并且缺失的这个圆点的位置在每一帧中的位置是按顺时针排列的。当你凝视画面中心的黑色十字几秒后，就会看到那个缺失的圆点变成了绿色的。

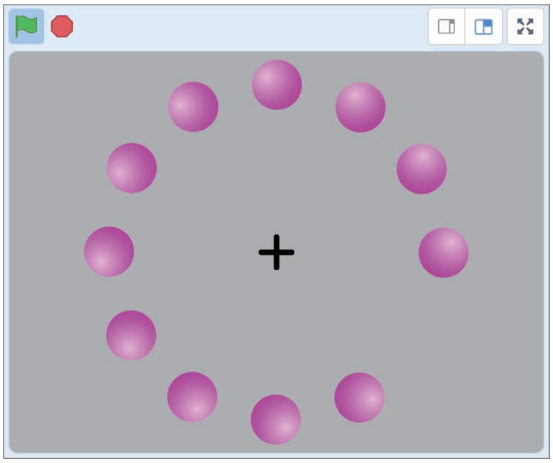

图 1-5-1　作品效果图

创作思路

按照时钟表盘的时针位置分布 12 个紫色的圆点，并且以 0.1 秒的时间间隔按照顺时针顺序隐藏其中的一个，即每次只显示 11 个紫色的圆点。

编程实现

先观看资源包中的作品演示视频 1-5.mp4，再打开模板文件 1-5.sb3 进行项目创作。

(1) 编写创建小球克隆体的代码(见图 1-5-2)。

在“创建小球克隆体”过程中，利用次数型循环和克隆积木创建 12 个小球角色的克隆体并设定其出现的位置。同时，从 1 开始给每个克隆体编号，将编号存放在私有变量 ID 中。

这 12 个小球克隆体被平均分布在距离舞台中心(0,0)位置 150 步的圆周上。即第 1 个小球克隆体从(0,0)面向 0 度方向移动 150 步，第 2 个小球克隆体从(0,0)面向 30 度方向移动 150 步，第 3 个小球克隆体从(0,0)面向 60 度方向移动 150 步，以此类推。

(2) 编写控制小球克隆体显示或隐藏的代码(见图 1-5-3)。

在“移动缺失圆点”过程中，通过“计数器”变量控制某个小球克隆体的显示或隐藏。“计

数器”变量的值以 0.1 秒的时间间隔从 1 到 12 不断地改变。

图 1-5-2　创建小球克隆体的代码

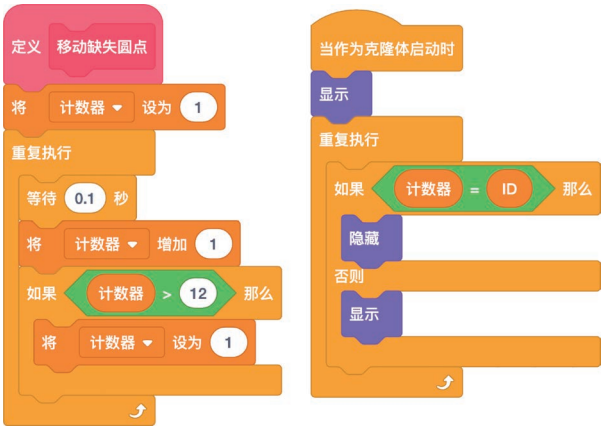

图 1-5-3　控制小球克隆体显示或隐藏的代码

另外，在“当作为克隆体启动时”积木下编写代码监听“计数器”变量值的变化。当某个小球克隆体的 ID 值与“计数器”变量的值相等时，则将该小球克隆体隐藏；否则，将其显示。

当项目运行后，可以切换到全屏模式，这样更方便观察呈现出的视错觉效果。另外，如果将作品中紫色的圆点换成其他颜色，那么缺失的圆点也将改变颜色。

1.6　消失的黄点

作品描述

该作品用于验证一种由运动引起的消失错觉，作品效果见图 1-6-1。当盯着舞台中间闪烁的绿色圆点几秒之后，就会看到位于三角形顶点处的 3 个静态的黄色圆点逐渐消失或重新出现。如果注意力非常集中，可以看到 3 个黄色圆点同时消失。

另外，作为背景的蓝色十字阵列在旋转时，有助于让黄色圆点快速消失。

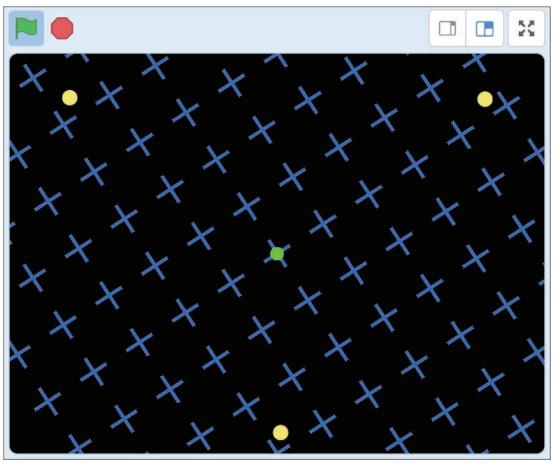

图 1-6-1　作品效果图

创作思路

该作品采用动画进行展示，生成“十”字图形阵列并控制其不停地旋转。具体方法是，先利用造型编辑器绘制一个蓝色的“十”字造型，然后利用马赛克特效积木生成“十”字图形阵列。

编程实现

先观看资源包中的作品演示视频 1-6.mp4，再打开模板文件 1-6.sb3 进行项目创作。

（1）创建“黄点”角色。

创建一个名为“黄点”的空角色，然后在该角色的造型编辑区中绘制 3 个黄色的圆点。两个圆点位于舞台上方左、右各一个，一个圆点位于舞台下方中间位置。通过属性面板将该角色定位在舞台中心位置，不需要编写代码。

（2）创建“绿点”角色并编写代码（见图 1-6-2）。

创建一个名为“绿点”的空角色，然后在该角色的造型编辑区中绘制一个绿色的圆点，大小可自行调整。

在“绿点”角色的代码区，利用“移到……”积木将“绿点”角色定位在舞台的中心位置，然后利用“重复执行”积木控制该角色以 0.5 秒的时间间隔不停地显示和隐藏。

（3）创建“十字”角色并编写代码（见图 1-6-3）。

图 1-6-2　控制“绿点”角色闪现的代码

图 1-6-3　生成“十”字阵列并旋转的代码

创建一个名为“十字”的空角色，然后在该角色的造型编辑区中绘制一个蓝色的“十”字图形，并将“十”字图形的四个末端拼接一段黑色的线段。之后，将造型大小调整为 150×150，并在角色属性面板中将角色大小设为 350。另外，需要将舞台的背景填充成黑色。

在“十字”角色的代码区，利用“将‘马赛克’特效设为 100”积木生成“十”字图形的阵列效果。然后，通过“重复执行”积木控制角色不停地旋转。

当项目运行后，可以切换到全屏模式，这样更便于观察呈现出的视错觉效果。

1.7　谁快谁慢

作品描述

该作品用于验证一种对运动快慢感知出错的视错觉，作品效果见图 1-7-1。假设有黄色

和蓝色的两个小车穿过黑白条纹的道路，它们其实是以相同的速度前进，但是会让你觉得其中一个总是慢了一步。当道路的黑白条纹被移除后，错觉消失，可以看到两个小车其实是同步前进的。

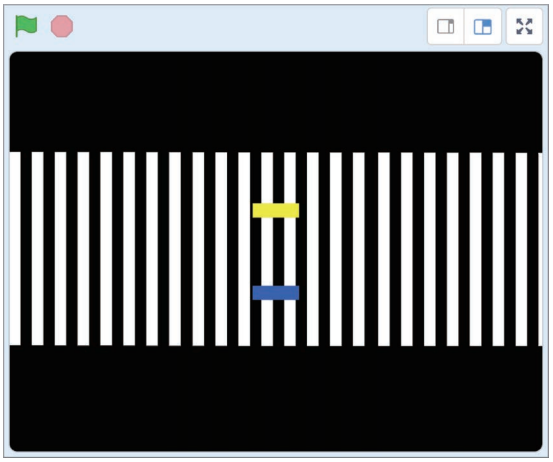

图 1-7-1　作品效果图

创作思路

创建一个包含黄色方块和蓝色方块的角色作为小车，并控制其在水平方向上平滑移动。另外，创建一个带有黑白条纹的角色作为道路，在小车行进过程中可以显示或隐藏道路。

编程实现

先观看资源包中的作品演示视频 1-7.mp4，再打开模板文件 1-7.sb3 进行项目创作。

(1) 创建“小车”角色和编写代码。

创建一个名为“小车”的空角色，然后在该角色的造型编辑区中绘制两个矩形代表小车，一个填充为黄色，一个填充为蓝色，两者大小相同，一上一下，垂直对齐。

如图 1-7-2 所示，这是控制“小车”角色移动的代码。在一个“重复执行”积木中，控制“小车”角色在 10 秒内从舞台左侧(－240,0)平滑移动到舞台右侧(240,0)，如此反复运动。

通过造型的设计和代码的编写，可以保证代表小车的两个矩形是同时移动的。

(2) 创建“道路”角色和编写代码。

创建一个名为“道路”的空角色，然后在该角色的造型编辑区中绘制两个造型，一个造型是由多个白色条纹构成，将其造型名称设为“有条纹”；另一个造型是灰色的且无条纹，将其造型名称设为“无条纹”。

如图 1-7-3 所示，这是切换不同道路造型的代码。当按下数字键 1 时，将“道路”角色的造型换成“有条纹”，舞台上显示的是一条带有黑白条纹的道路；当按下数字键 2 时，将“道路”角色的造型换成“无条纹”，舞台上显示的是一条灰色的道路。

当项目运行后，通过按下数字键 1 或 2 切换不同的道路，观察舞台上两辆小车的运动情况，看看谁快谁慢。