

变异的吃豆人（单片机版）实习报告

鲁申怡 金子路

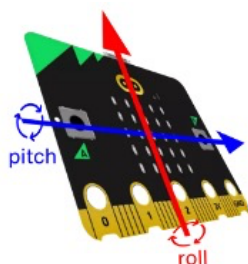
元培学院

摘要：以micro:bit编程经典案例“重力控制水平仪”为基础，参考经典小游戏吃豆人，我们对游戏规则和游戏呈现方式稍作改变，设计了这款变异的吃豆人（单片机版）小游戏。在原版游戏的基础上，我们为玩家提供不同游戏难度选项，并且对“目标点”的设置有一定修改。受限于单片机的“小巧”，这款游戏尚有不足，期待进一步完善。我们希望变异的吃豆人（单片机版）可以向大家展示micro:bit编程的神奇魅力，并向经典游戏“吃豆人”致敬。

1 选题及创意介绍

利用micro:bit设计重力控制水平仪是micro:bit编程的经典案例之一。借助micro:bit上的加速度计，可以测量上下（纵向）以及左右（横向）的倾斜（如图），从而实现一个重力控制上下左右滑动的LED，将单片机的倾斜角度转化为显示面板上LED灯的位置，非常直观。

由此，我们设想可以在micro:bit重力控制水平仪的基础上，与简单小游戏“吃豆人”相联系，设计一款通过改变单片机倾斜角度，控制“吃豆人”位置，在单片机控制面板上吃掉“目标点”获得得分的“变异的吃豆人（单片机版）”小游戏。



经过一系列天马行空的讨论，我们对“变异的吃豆人（单片机版）”小游戏有如下规则制定（说明书不容易读懂，建议看视频演示或者直接尝试）：

在一个神秘的迷宫中，住着一群拥有特殊能力的小生物，他们的身体由黄色的圆形和三条黑线组成，就像豆子一样。他们被称为“吃豆人”，因为他们的主要任务就是寻找和吃掉所有的豆子。吃豆人生活在一个充满危险的世界里，他们需要不断地躲避各种障碍和敌人的袭击，才能顺利地完成任务。但是，他们拥有非凡的速度和机智，可以通过不同的技巧和战术来应对各种挑战。每当他吃掉一个豆子，就会增加一点能量和力量，让他更加强大。

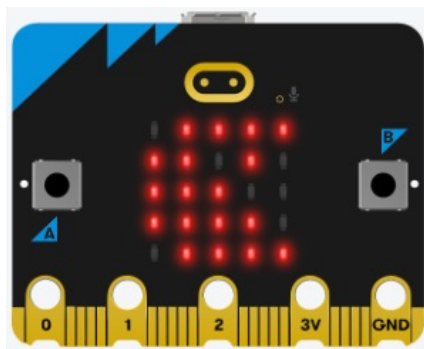
- 1.程序初始化完成后，按下按钮A可以开始游戏，触摸金属Logo可以更改游戏难度，游戏难度从低到高为1-5，默认为1，直接按下A开始游戏的难度为1；
- 2.通过改变单片机倾斜角度，控制“吃豆人”的位置；
- 3.游戏过程中若按下按钮B，游戏强制终止；
- 4.每吃掉一个“目标点”，也即“吃豆人”与“目标点”位置重合，得分将+1；
- 5.初始时，单片机上将显示3个“目标点”，每吃掉2个“目标点”后，单片机显示面板上将再次生成2个“目标点”，使显示面板上“目标点”总数再次变为3个；
- 6.得分达到8分后，显示面板上将出现“Killer（地雷）”，且“Killer”数量随得分增加而增加；
- 7.每间隔一定时间，“Killer”位置将改变，游戏难度越高，Killer移动速度越快；

- 8.若踩到“地雷”，也即“吃豆人”与“地雷”位置重合，则游戏结束，“U r dead”，单片机显示面板将显示玩家得分。若想再次游戏，按下按钮A。
- 9.若得分超过30，则游戏成功，“U win”，单片机面板将显示玩家得分。若想再次游戏，按下按钮A。
- 10.由于单片机显示面板只能显示红色，我们用亮度区分“吃豆人”，“目标点”和“地雷”。其中“吃豆人”最亮，“目标点”次亮，“地雷”最暗。

2 设计方案和硬件连接

我们考虑并调用了MicroBit中的加速度传感器、扬声器、LED点阵模块，并利用了其自带的按钮与Logo触摸板。通过侦测X与Y方向的加速度分量，来控制Player的移动，并将Player、Killer、Goal以不同亮度在LED上显示出来。对于Killer而言，我们设计了Killer追逐Player的相关算法，并为该算法加入了随机函数，使得每一个Killer的每次追逐Player的过程是相对随机的，从而使玩家难以找到Killer移动的规律。

我们的初版方案并未加入移动的Killer（因此最初把Killer称为Bomb）。后续的不断完善中，我们为该程序加入了Killer的追逐算法，为游戏带来更多乐趣（也导致了难度上升，可能难以通关），同时也加入了难度的选择界面，不断完善了游戏中可能遇到的如中途退出的情况，以求提供更完整的游戏体验。



3 实现方案及代码分析

我们将“吃豆人”设置为player，“目标点”设置为goal，“地雷”设置为bomb。

第一重While循环用于检测button_a是否被按下，若没有按下，始终显示箭头提示图案；若按下，开始第二重While循环。

第二重While循环首先调用movecontrol()，每次循环都基于加速度计探测到的数值重置player的位置。再利用条件判断if语句，分别调用函数。

一局游戏结束后，退出第二重While循环，重新进入第一重While循环。

关于实现过程和更加详细的代码分析可以参看下文代码中的注释，我们相信已然比较明晰。

```
1  # Imports go at the top
2  from microbit import *
3  import random, music, speech, time
4
5  start = False
6  Exit = True
7  points = 0 #初始得分0
8  panel = []
```

```

9  # 由于单片机只能显示红色，可以用亮度区分目标点，player点和地雷
10 # 设定player点亮度9，目标点亮度6，地雷亮度3
11 pl = 9 #player亮度playerlight
12 gl = 6 #goal亮度goallight
13 bl = 3 #bomb亮度bomblight
14 interval = [4000, 3000, 2000, 1000, 500]
15 level = 0
16
17 #单片机上显示所有点的坐标
18 lst = [[0,0],[0,1],[0,2],[0,3],[0,4],[1,0],[1,1],[1,2],[1,3],
19         [1,4],[2,0],[2,1],[2,2],[2,3],[2,4],[3,0],[3,1],[3,2],
20         [3,3],[3,4],[4,0],[4,1],[4,2],[4,3],[4,4]]
21 num =
22     ['1234','1324','1342','1423','1432','1243','2134','2143','2314','2341','2413','2431',
23
24     '3124','3142','3214','3241','3412','3421','4123','4132','4213','4231','4312','4321']
25
26 player = [2,2]
27 goalset = []
28 bombset = []
29 bombTime = time.ticks_ms()
30
31 #控制player随加速度变化移动位置
32 def movecontrol():
33     global player
34     '''每次刷新player点位置，先将上一轮的player亮度设置为0，再重新设置新一轮的player亮
35     度'''
36     display.set_pixel(player[0],player[1],0)
37     x = (accelerometer.get_x()+1000)//400
38     y = (accelerometer.get_y()+1000)//400
39     if x > 4:
40         x = 4
41     if y > 4:
42         y = 4
43     if x < 0:
44         x = 0
45     if y < 0:
46         y = 0
47     player = [x, y]
48     display.set_pixel(x, y, pl)
49
50 #随机生成一个目标点并将目标点放进goalset
51 def setgoal():
52     global panel
53     goalindex = random.randint(0, 24)

```

```

51     currentgoal = lst[goalindex]
52     if (currentgoal not in goalset and currentgoal not in bombset
53         and currentgoal != player):
54         '''保证新加入的目标点和player, 地雷不重合'''
55         display.set_pixel(currentgoal[0], currentgoal[1], gl)
56         goalset.append(currentgoal)
57         panel[currentgoal[0]+1][currentgoal[1]+1] = 1
58     else:
59         setgoal()
60
61 #随机生成一个地雷并将地雷放进bombset
62 def setbomb():
63     global bombset
64     bomb = random.randint(0,24)
65     while ((lst[bomb] in goalset) or (lst[bomb] in bombset) or
66         (lst[bomb] == player)):
67         '''保证新加入的地雷与先前的player, 目标点都不重合'''
68         bomb = random.randint(0,24)
69     bombset.append(lst[bomb])
70     display.set_pixel(lst[bomb][0], lst[bomb][1], bl)
71
72 #地雷 (怪物) 追逐玩家
73 def bombmove():
74     global bombTime, bombset, panel, num, interval, level
75     currentTime = time.ticks_ms() #记录时间
76     dir = [[1,0],[-1,0],[0,1],[0,-1]] #移动放下: 上下左右
77     if time.ticks_diff(currentTime, bombTime) >= interval[level]: #判断时间间隔
78         for i in range(len(bombset)):
79             order = random.choice(num) #随机设置移动顺序, 处理bomb与goal重合的情况
80             bomb = bombset[i] #选取一个bomb
81             display.set_pixel(bomb[0], bomb[1], 0) #将上一轮的该bomb的亮度调0
82             XorY = random.randint(0,1) #XorY=0在x方向移动; XorY=1在y方向移动
83             xDst = player[0] - bomb[0] #判断与player在x方向的距离
84             yDst = player[1] - bomb[1] #判断与player在y方向的距离
85             if xDst == 0: #如果已经和player的x坐标值相同, 就在y方向移动
86                 XorY = 1
87             elif yDst == 0: #如果已经和player的y坐标值相同, 就在x方向移动
88                 XorY = 0
89             if XorY == 0 and xDst > 0: #在x方向移动
90                 if [bomb[0]+1, bomb[1]] in goalset: #如果移动后的位置和goal重合
91                     for d in order: #order是随机生成的移动顺序
92                         d = int(d) - 1
93                         if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
94                             bomb = [bomb[0]+dir[d][0], bomb[1]+dir[d][1]]
95                             break

```

```

96         else:
97             bomb[0] += 1
98         elif XorY == 0 and xDst < 0: #类比
99             if [bomb[0]-1, bomb[1]] in goalset:
100                 for d in order:
101                     d = int(d) - 1
102                     if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
103                         bomb = [bomb[0]+dir[d][0], bomb[1]+dir[d][1]]
104                         break
105             else:
106                 bomb[0] -= 1
107         elif XorY == 1 and yDst > 0: #类比
108             if [bomb[0], bomb[1]+1] in goalset:
109                 for d in order:
110                     d = int(d) - 1
111                     if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
112                         bomb = [bomb[0]+dir[d][0], bomb[1]+dir[d][1]]
113                         break
114             else:
115                 bomb[1] += 1
116         elif XorY == 1 and yDst < 0: #类比
117             if [bomb[0], bomb[1]-1] in goalset:
118                 for d in order:
119                     d = int(d) - 1
120                     if panel[bomb[0]+dir[d][0]+1][bomb[1]+dir[d][1]+1] == 0:
121                         bomb = [bomb[0]+dir[d][0], bomb[1]+dir[d][1]]
122                         break
123             else:
124                 bomb[1] -= 1
125         print(bomb)
126         display.set_pixel(bomb[0], bomb[1], b1)
127         bombset[i] = bomb
128         bombTime = time.ticks_ms()
129
130 #初始化设置
131 def initial():
132     global goalset, panel
133     panel = [[1,1,1,1,1,1,1],
134              [1,0,0,0,0,0,1],
135              [1,0,0,0,0,0,1],
136              [1,0,0,0,0,0,1],
137              [1,0,0,0,0,0,1],
138              [1,0,0,0,0,0,1],
139              [1,1,1,1,1,1,1]
140             ]

```

```

141     display.set_pixel(player[0], player[1], pl)
142     i = 0
143     while i < 3:
144         '''初始时随机生成三个目标点'''
145         index = random.randint(0, 24)
146         if lst[index] not in goalset and lst[index] != player:
147             '''保证三个目标点各不相同而且和player不重合'''
148             goalset.append(lst[index])
149             panel[lst[index][0]+1][lst[index][1]+1] = 1
150             display.set_pixel(lst[index][0], lst[index][1], gl)
151             i += 1
152
153 #当player与某一个目标点重合调用eat()
154 def eat():
155     global goalset, player, points
156     music.play(['C5:1']) #声音提示吃到目标点
157     #print(player) #打印吃掉的点, 在调试代码时帮助很大
158     goalset.remove(player)
159     panel[player[0]+1][player[1]+1] = 0
160     points += 1
161
162 #当player与某一个地雷重合调用dead()
163 def dead():
164     music.play(['D4:1']) #声音提示踩到地雷
165     '''结束流程: 显示骷髅头, speech嘲讽, 滚动U r dead, 显示得分'''
166     display.show(Image.SKULL)
167     sleep(400)
168     speech.say('ha, ha, You are Dead')
169     display.scroll('U r dead', delay=80)
170     display.scroll('Got')
171     display.scroll(points)
172     sleep(2000)
173     display.clear()
174
175 #当得分超过30, 本轮游戏胜利, 调用win()
176 def win():
177     music.play(music.PRELUDE, wait=False) #在悦耳的音乐中迎接胜利吧
178     '''结束流程: 显示笑脸, 滚动U win, 显示得分'''
179     display.show(Image.HAPPY)
180     sleep(2000)
181     display.scroll('U win', delay=80)
182     display.scroll('Got')
183     display.scroll(points)
184     sleep(2000)
185     display.clear()

```

```

186
187 #按pin_logo选择难度，调整bomb移动的时间间隔
188 #难度默认为最低=1，每按一次难度+1
189 def levelAdjust():
190     global interval, level
191     if pin_logo.is_touched():
192         music.play(['E4:1'])
193         level += 1
194         if level > 4:
195             level = 0
196         display.show(level+1)
197
198 music.play(music.RINGTONE, wait=False) #欢迎音乐
199 display.show(Image.PACMAN)
200 sleep(500)
201 while Exit:
202     levelAdjust()
203     if button_a.was_pressed():
204         movecontrol()
205         display.clear()
206         bombset = []
207         goalset = []
208         points = 0
209         initial()
210         start = True
211         Exit = False
212         #setbomb() #此行供测试使用
213     while start:
214         movecontrol()
215         bombmove()
216         if button_b.was_pressed():
217             #按button_b可以强制退出，显示已有得分
218             display.scroll('Got')
219             display.scroll(points)
220             sleep(500)
221             goalset = []
222             bombset = []
223             start = False
224             Exit = True
225             display.clear()
226             display.show(Image.PACMAN)
227             break
228         if player in goalset: #吃到目标点
229             eat()
230             print(points)

```

```
231         if player in bombset: #踩到地雷
232             start = False
233             dead()
234             Exit = True
235             display.show(Image.PACMAN)
236             break
237     if len(goalset) == 1: #如果吃掉了两个目标点，就更新goalset
238         setgoal()
239         setgoal()
240     if points//8 - len(bombset): #每多吃掉8个目标点，增设一个地雷
241         setbomb()
242         bombTime = time.ticks_ms()
243     if points == 30: #points达到30，成功通关
244         win()
245         Exit = True
246         start = False
247         display.show(Image.PACMAN)
248         break
```

4 后续工作展望

由于单片机上显示面板的大小有限，“地雷”数量不能无限增多，得分达到30即可通关。而且单片机上只能显示红色的LED灯，不利于区分“吃豆人”、“目标点”和“地雷”。后续希望能将单片机和其他显示工具结合起来，在更大且呈现颜色更多的显示器上呈现游戏，提高游戏难度上限，带给玩家更好体验和更多刺激。

5 小组分工合作

源代码由金子路和鲁申怡共同完成。Poster和运行和介绍视频由金子路制作，实习报告主要由鲁申怡主笔。但实际上，整个小游戏是两个人共同构思、共同创作的结果。我们共同合作，互相帮忙debug，才能将这款变异的吃豆人（单片机版）呈现给大家。