学生作品选编

关于使用 TI-83 Plus 图形计算器制作动画效果的探究学习

上海市南汇中学　严晨宇（高一学生）

（指导老师　凤 杰）

TI 图形计算器与其他科学计算器有很大的不同，其中最大的不同便是 TI 图形计算器强大的“作图”功能。而如何让图象动起来而成为“动画”，对此我作了一些自主的探究学习。

我的探究性学习是很自由、很自主的。无论是探究学习的方向、方式，还是方法完全由我自己决定。由于我缺乏计算机编程方面的知识，所以采用了多实验、多参考说明书、多与同学和老师交流、讨论的方法。

在此过程中，我也走过不少弯路。但最终还是找到了自认为只要精心设置便可制作任意好用的动画效果的一种方法。

最初，我翻看 TI 操作说明书时就发现其中有“动画”一词——“　编辑器”中的一种绘图方式。后来寻找资料，又知道简单动画主要是由运动对象（后面简称“动元”）和运动路径两者构成的。有一次，我又恰好看到了由老师提供的外校同学制作的投篮图（不过是图片而非动画）。既然是“投篮”，那怎么让篮球运动起来呢？我想到了第一种方法：利用函数的绘图方式的方法，即使用函数模拟图象和运动路径，再用绘图方式中的动画效果便可制作。示例：

图 1

1、运行模式、图象样式和窗口设置如图 1

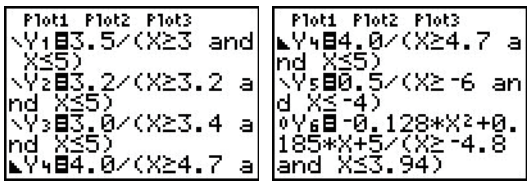


图 2

2、在[Y=]编辑器中输入模拟函数，其中Y_6的绘图方式改为“”动画模式（如图 2）

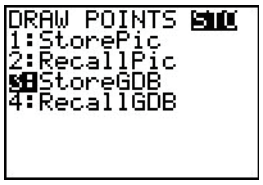


图 3

3、为了方便起见，可将以上所有设置存入 GDB 变量（图象数据库），并用 RecalGDB 命令直接调用即可（图 3），执行效果如下（组图 4，应该是连续显示）：

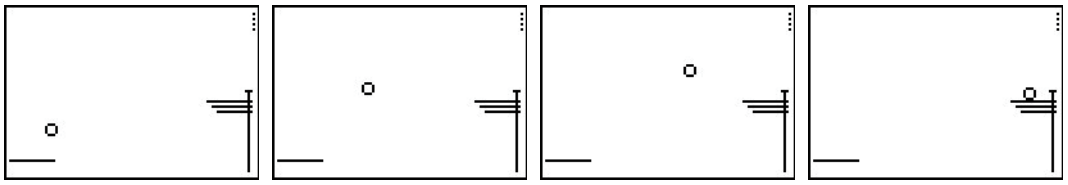


图 4

此法优点在于动画效果较好，而且因为直接调用命令，所以制作也并不太难。但是有两个致命缺点：

（1）运动物体只能是小球；

（2）必须利用函数解析式得到图象，与直接作图模式（Draw）不兼容。

而这两点，都使动画本身得不到普及，足以显出无法克服的弱点。

后来，我想到 TI-83 Plus 自带的视频小游戏，它能让我们爱不释手，关键在于里面的图象十分有趣，人机交流做得也很不错。所以我觉得：动画效果绝不是那么简单的。于是我询问了老师，从老师那里得到了一个重要提示：动画效果其实是由图片一幅一幅地不断切换而做成的，与电影胶片的放映原理一样。因此我想到了“DRAW 菜单”下的“STO 菜单”,因为计算器可以存储图片，然后使图片不停地自动更换不就可以了么！在这一想法下，运用计算器的编程功能，我尝试制作了名为“Face”的程序:

```
PROGRAM:FACE
:ClrDraw
:For(J,1,25)
:RecallPic 1
:End
:ClrDraw
:For(J,1,25)
:RecallPic 2
:End
:ClrDraw
:For(J,1,25)
:RecallPic 3
:End
:ClrDraw
:For(J,1,25)
:RecallPic 4
:End
:ClrDraw
:For(J,1,25)
:RecallPic 5
:End
:ClrDraw
:For(J,1,25)
:RecallPic 6
:End
:ClrDraw
:For(J,1,25)
:RecallPic 7
:End
```

```
PROGRAM:FACE
:ClrDraw
:For(J,1,25)
:RecallPic 8
:End
:ClrDraw
:For(J,1,25)
:RecallPic 9
```

```
PROGRAM:FACE
:RecallPic 8
:End
:ClrDraw
:For(J,1,25)
:RecallPic 9
:End
:
```

说明：其中 Pic1—9 为预先存入的图片，然后用 ClrDraw 和 RecalPic 不断进行擦除、调用即可（当时还没有加入“For……End 空循环语句），程序运行后出现如下变脸效果（图 5）。

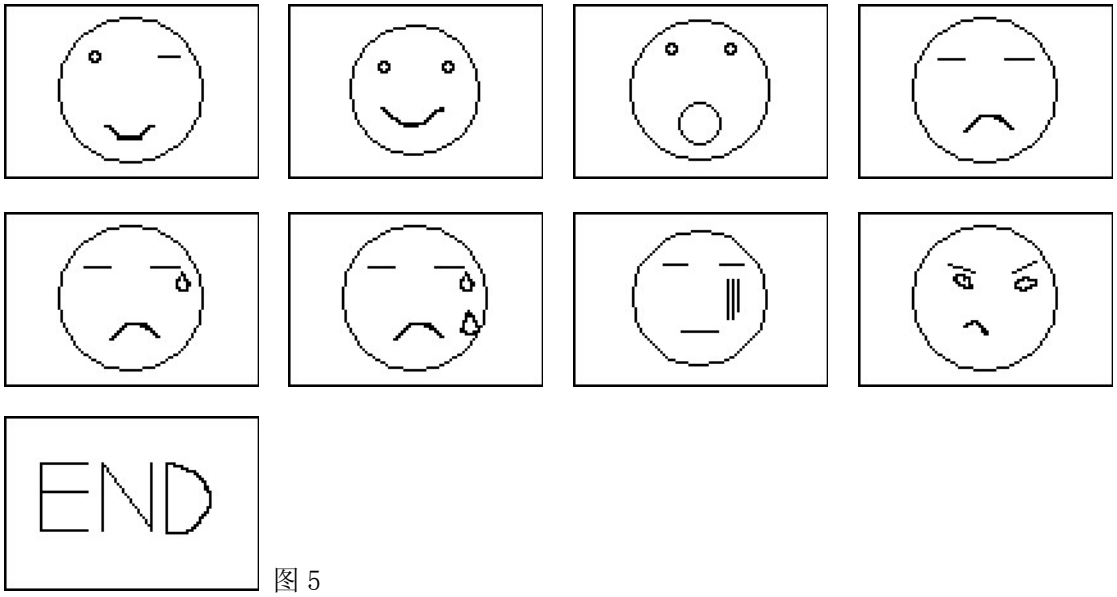


图 5

这种方法与最初的方法相比又有几个优点：（1）动画是全屏显示的；（2）“动元”突破了小球的限制；（3）运动路径可以随意。但是仅仅如此仍解决不了几个技术难点。首先，图片占的内存很大，所以存储数量有限（最多 10 张）；其次，刚编程序时我遇到了一个几乎无法解决的问题：执行的速度太快，10 张图几乎是一闪而过，根本看不清过程。

所以，我决定寻找更好的动画制作方法。几番努力，在研究计算机的 Output 功能时，我想到可以利用一点作为物体运动的质点（物体形状可以在这点上构造），并在点的坐标中引入两个变量（A，B），通过变量 B 与 A 之间的函数关系来决定运动路径，再通过 For 循环结构来不时改变变量 A 值，结合“ClrDraw”命令，便能产生数量大而连贯的图片，从而合成一个较完整的动画效果。这种方法与前两种方法相比有较大的改善，主要克服了第一种方法中“动元”形状和第二种方法中显示帧数的限制。在投篮动画的基础上我改掉了小球（可以画其他形状的几何体，这里用矩形演示），制作了“动元”沿抛物线运行的动画轨迹：

（一）程序名“FLYBOX”（飞盒）：

```
PROGRAM:FLYBOX
:ClrDraw
:0→B
:For(A,-6,4,0.5)

:For(D,1,10)
:Line(A,B,A,B+1)
```

```
PROGRAM:FLYBOX
:Line(A,B+1,A+1,
B+1)
:Line(A,B,A+1,B)

:Line(A+1,B,A+1,
B+1)
:Line(-6,0,-5,0)
```

```
PROGRAM:FLYBOX
:Line(4,0,5,0)
:1+D→D
:End
:-0.2083*A²-0.41
6*A+5→B
:ClrDraw
:End
```

运行效果（组图 6）：

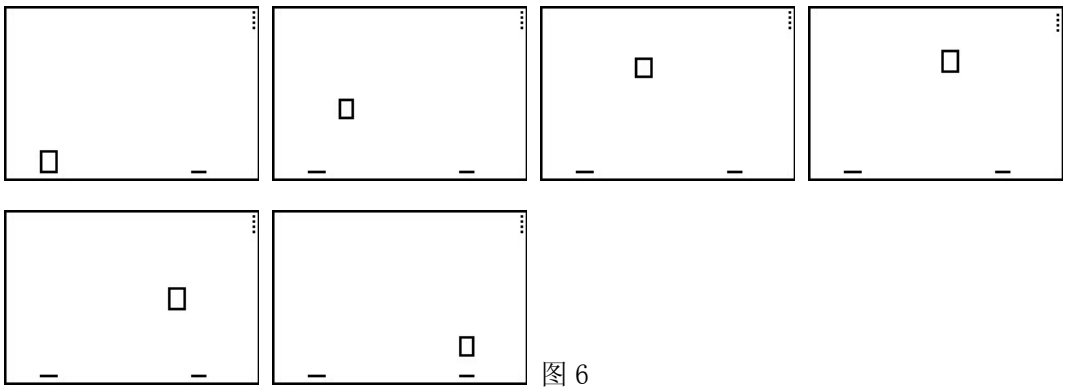


图 6

在此之前，我想过直接使用 “circle”作圆来表示篮球。但是由于圆是慢慢作出的，所以效果很不真实；而且当时也没想到用“ClrDraw”命令来擦除图象，所以一时之间只能放弃。

另外对于前面提到的因为执行太快而闪烁的问题，很长一段时间内得不到解决。直到后来，在使用循环结构时，我发现它执行起来要一定的时间。于是我使用了一种“空循环”来延迟时间，使得动画效果有较大的提高。通过改变 For 语句中参数值的大小可以决定延迟时间，同样方法也可以决定循环执行的次数，这样动态显示的速度（包括间隔和帧数）就可以任意控制了。

又如较能体现完整效果的“射箭”动画：

（二）程序名“ARROW”（射箭）：

```
PROGRAM:ARROW
:ClrDraw
:3→S
:While S≤6
:RecallPic 0
:For(T,1,4)
:Line(3,8,S,4)
:Line(3,0,S,4)
```

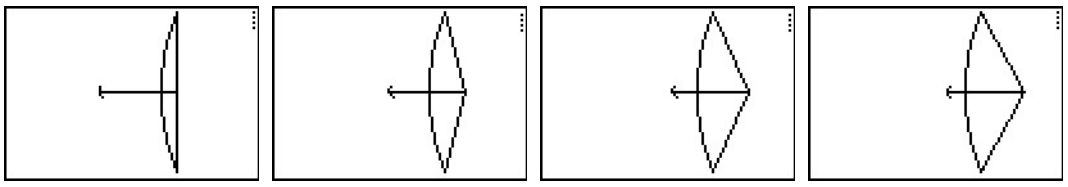
```
PROGRAM:ARROW
:Line(S-5,4,S,4)
:Line(S-4.9,3.8,
S-5.1,4)
:Line(S-5,4.2,S-
5.1,4)
:End
```

```
PROGRAM:ARROW
:0.5+S→S
:ClrDraw
:End
:While S≥-6
:For(T,1,5)
:Line(S-5,4,S,4)
```

```
PROGRAM:ARROW
:Line(S-5.2,4,S-
4.8,4.2)
:Line(S-5.2,4,S-
4.8,3.8)
:End
:S-1→S
:ClrDraw
```

```
PROGRAM:ARROW
:End
:
:
:
:
:
```

运行效果（组图 7）：



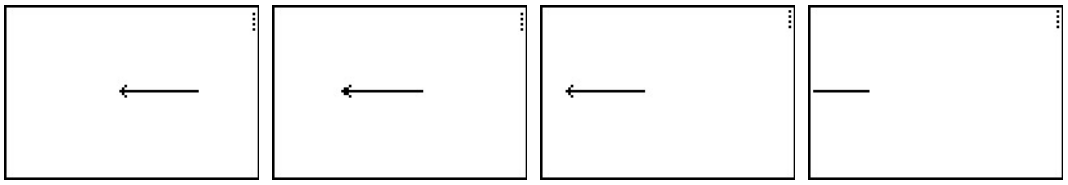


图 7

【注】两个程序的运行模式、图象样式和窗口设置均如图 8 所示，可以如图 3 存入某一“GDB #”变量并在程序运行前调用，或者在程序开头加入命令行“RecalGDB #”。

图 8

存在问题

我不清楚还有没有更好的方法，而且最后一种制作方法也存在很大缺陷，甚至与 TI-83 图形计算器里随机附带的几个小游戏中类似的动画效果仍有很大差距，主要有几点：

（1） 目前我只限于用点、线、英文字母作为动画元素，但是理论上只要有足够的耐心和良好的美术功底就可以画出更美观的“动元”；

（2） “动元”与“不动元”还无法区分开来，运动对象不能叠加到除了点、线以外的其他背景图象上，比如投篮动画中篮框、篮架应该是不动的；

（3） 运动路径能否脱离函数关系变成任意路径，其中对于计算器的获取键位功能（GetKey 命令）还不大能熟练运用（键位获取的意义在于可以在动画过程中通过光标键或其他键来指定下一个动作，从而达到任意运动的目的，我想那些游戏大概就是这样做出来的）；

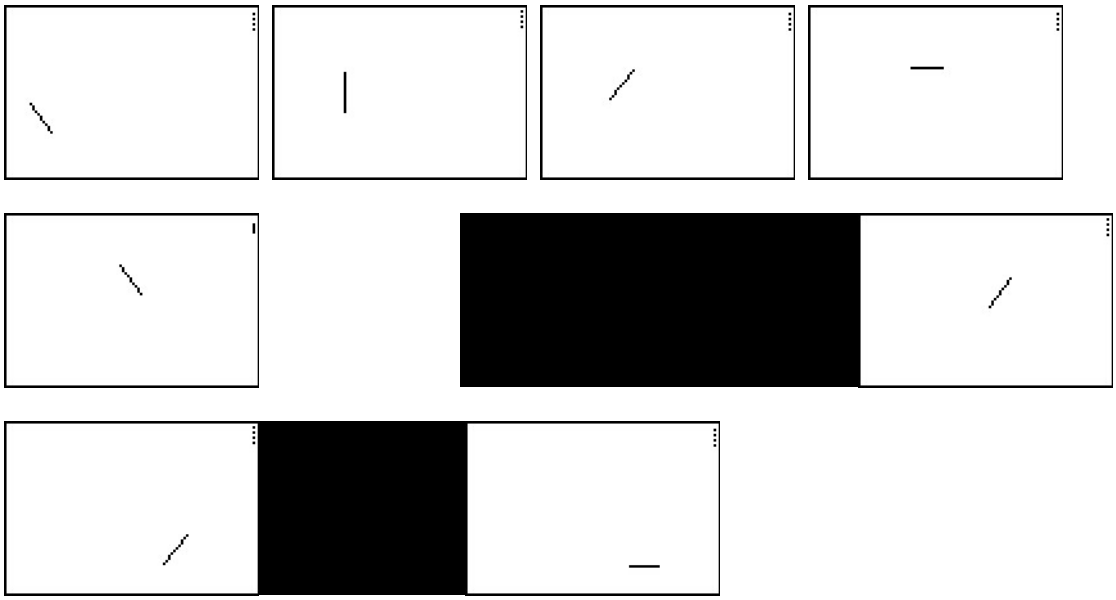
还有我考虑能否实现更复杂一些的动画，比如导弹沿抛物线发射的同时弹头方向也要随时作调整等等。象这样物体在移动中本身还做旋转变化的动画过程（所谓的二维变化）老师说其实也可以做到的，这就更激发了我进一步探究的欲望，我相信通过努力与尝试我会实现的。一个初步设想是增加角度旋转量θ，用线段简单演示如下：

程序

```
PROGRAM:B
:π/4→θ
:-6→A
:While A≤4
:For(J,1,10)
:Line(A+cos(θ),B
-sin(θ),A-cos(θ)
,B+sin(θ))
```

```
PROGRAM:B
:End
:0.5+A→A
:-0.2083*A²-0.41
6*A+5→B
:π/4+θ→θ
:ClrDraw
:End
```

运行效果：



学习体会

在做完了这方面探究后，回顾自己前前后后尝试制作的各个作品，我感到这次学习的收获是比较大的，探究的过程令我受益匪浅。

首先它让我更加明白，我所做的探究不一定是他人未知的、不曾做过的，毕竟优秀的动画作品数不胜数，而我做的却很粗糙。我们所做的研究学习很多时候可以、甚至必须站在前人或别人的研究基础上确立自己的目标，就象我在 TI 计算器平台上开发动画效果一样。这其实在我看来决不是不道德，而正是我们的文明、技术继承和发展的基础。

其次，让我感受颇深的是：永远不要太满足取得的成就，并且在取得阶段性成功后不要花力气去掩盖缺点。前一句是人们成千上万遍地说过的话。在我做的时候，除了前面所述的三个阶段出现 “想到”与“想不到”之外，有时便有一些满足——其实如果多用点心思直接用函数的动画效果也能做出不错的动画，用图片法同样也可以。这些想法往往容易阻碍我的前行，而这也是我所说的不要“掩盖缺点”的一例明证。如果有一点小小突破，但缺陷仍不小，就不应当花心思弃而转投他法来弥补这种缺陷。这会花费很多精力，还会让人满足。

最后一点感受，是做过一点小研究的人的共同体会，就是要敢想，要多试、要多求教于人。这样不仅能得到实质上的帮助，更重要的是能出其不意的带来灵感。

参考文献：〈TI-83 Plus Texas Instruments〉操作说明书