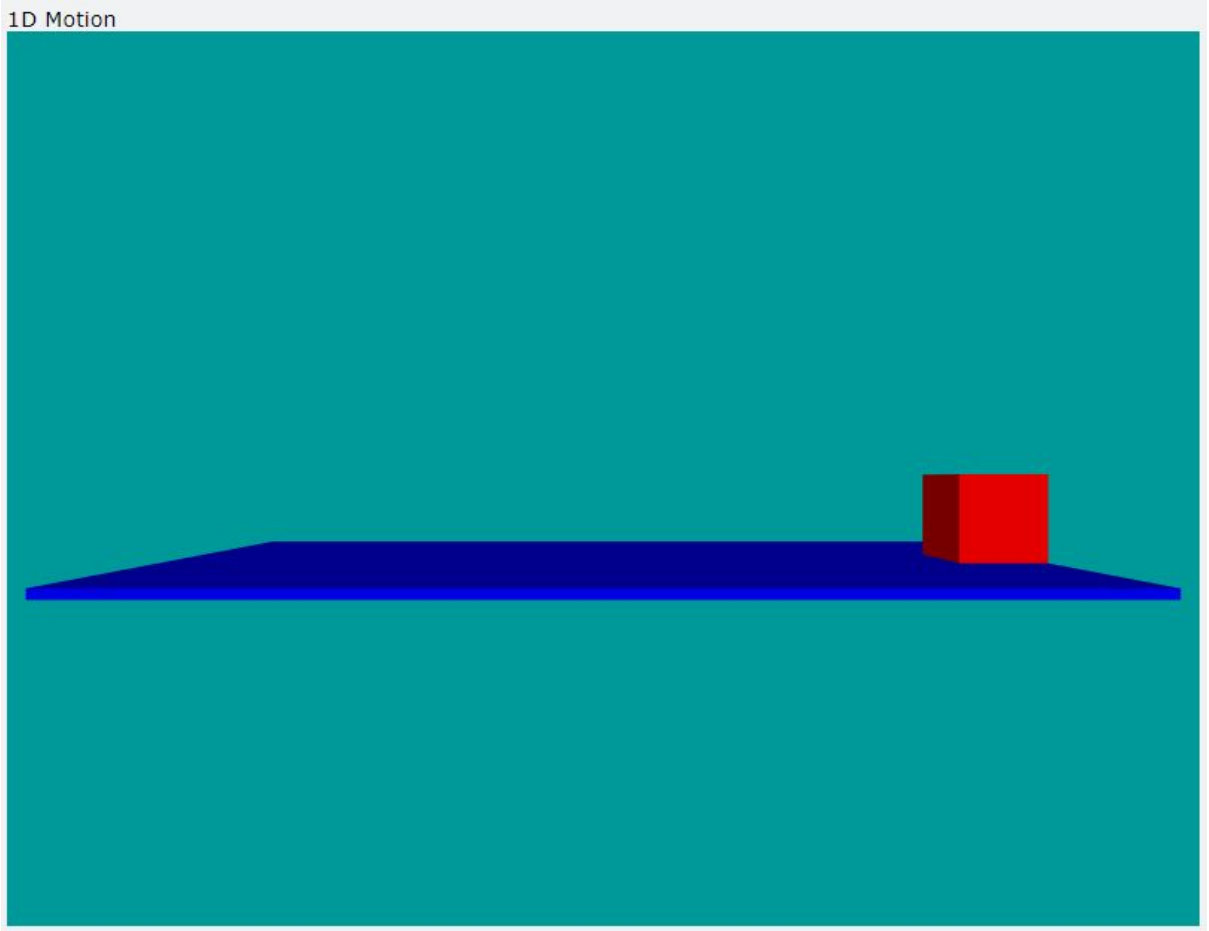
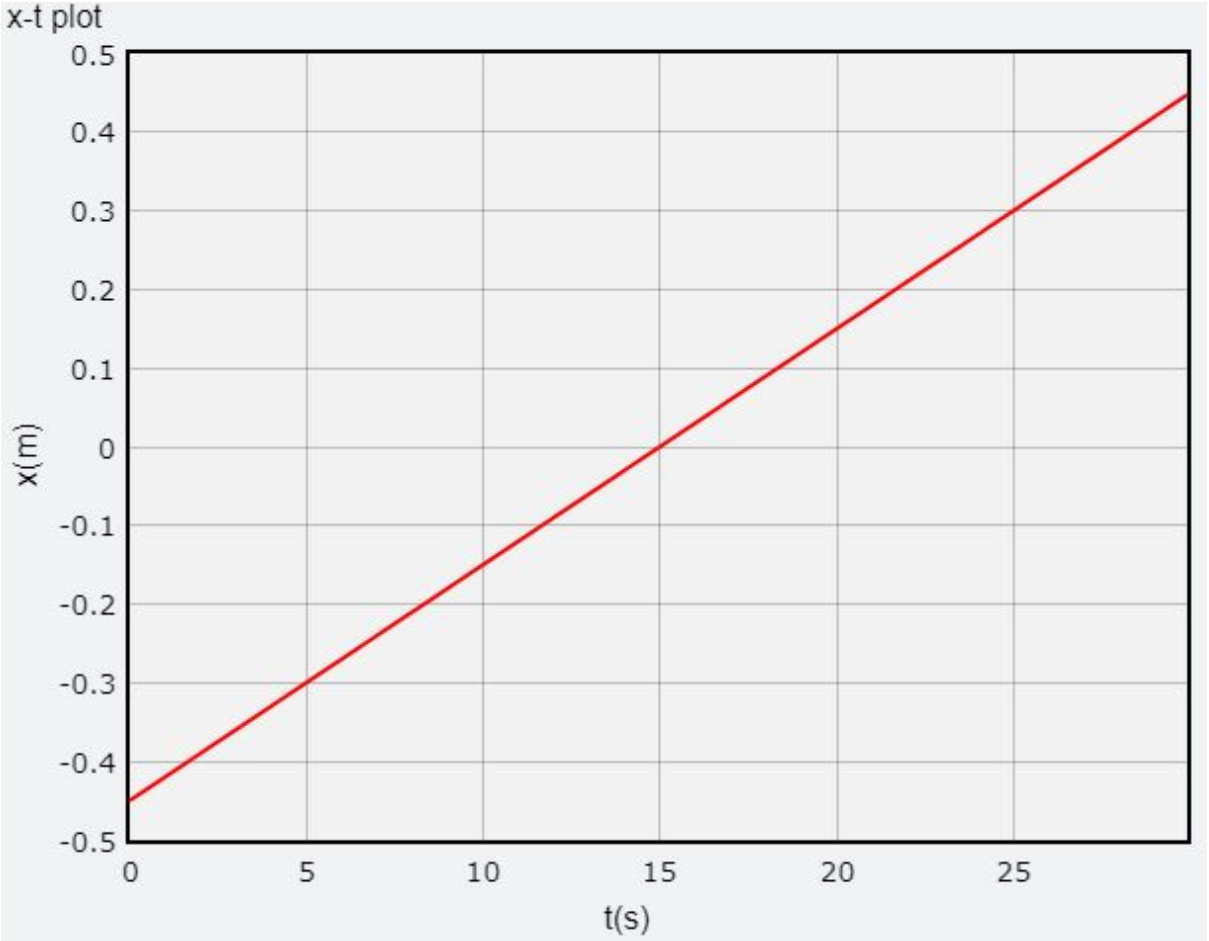
等速度直線運動

日期：2018/3/7

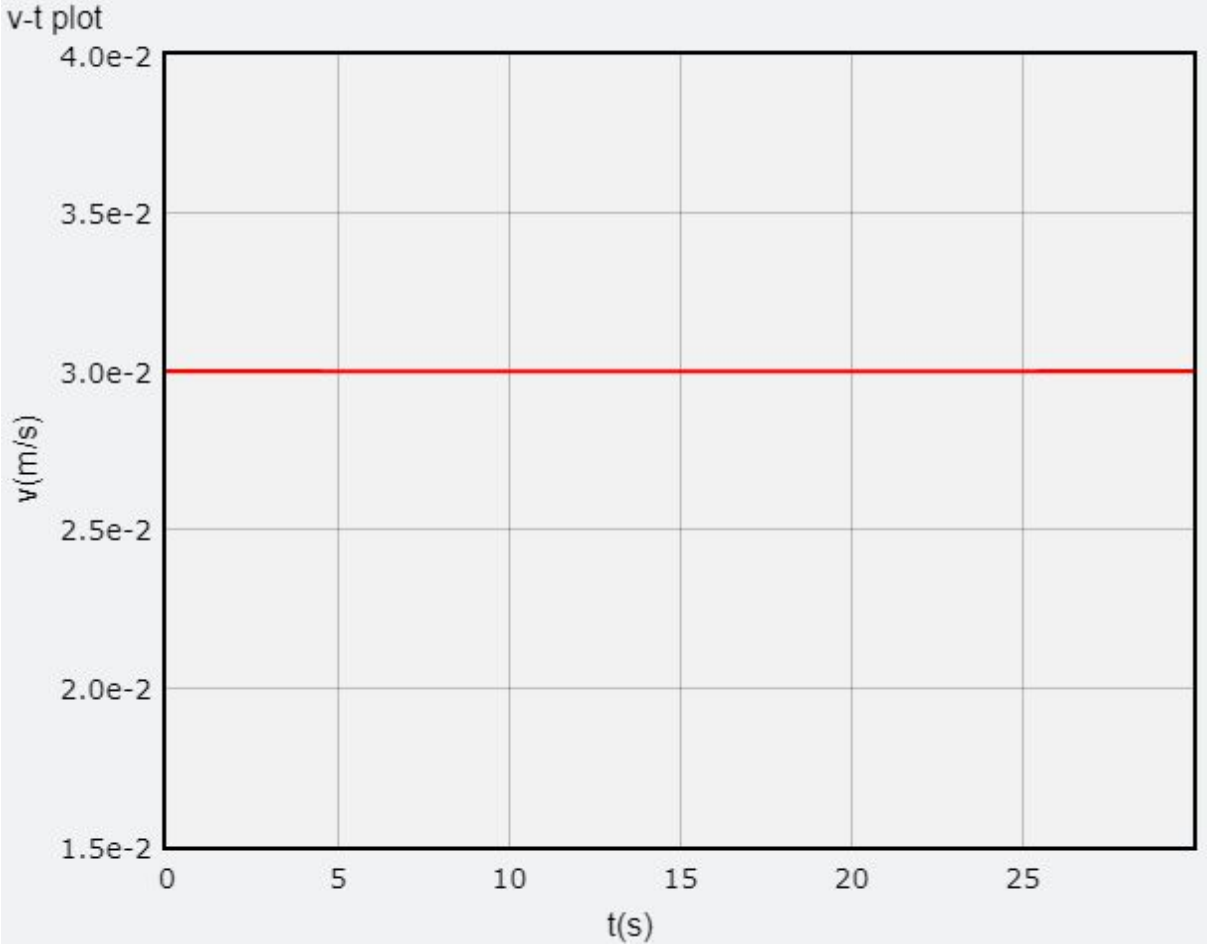
終於要開始做物理模擬動畫，我們先從最單純的等速度直線運動開始，目標是畫出木塊、地板以及木塊的 *x-t* 圖、 *v-t* 圖，成果如下：



等速度直線運動畫面截圖



木塊的 *x-t* 圖



木塊的 *v-t* 圖

程式 3-1：等速度直線運動　（取得程式碼）

```
"""
 VPython教學: 3.等速度直線運動
 日期: 2018/2/18
 作者: 王一哲
"""
from vpython import *

"""
 1. 參數設定, 設定變數及初始值
```

```
"""
size = 0.1    # 木塊邊長
L = 1         # 地板長度
v = 0.03      # 木塊速度
t = 0         # 時間
dt = 0.01     # 時間間隔

"""
 2. 畫面設定
"""
scene = canvas(title = "1D Motion", width = 800, height = 600, x =
0, y = 0, center = vector(0, 0.1, 0), background = vector(0, 0.6,
0.6))
floor = box(pos = vector(0, 0, 0), length = L, height = size*0.1,
width = L*0.5, color = color.blue)
cube = box(pos = vector(-L*0.5 + size*0.5, size*0.55, 0), length =
size, height = size, width = size, color = color.red)
cube.v = vector(v, 0, 0)
gd = graph(title = "x-t plot", width = 600, height = 450, x = 0, y =
600, xtitle = "t(s)", ytitle = "x(m)")
gd2 = graph(title = "v-t plot", width = 600, height = 450, x = 0, y
= 1050, xtitle = "t(s)", ytitle = "v(m/s)")
xt = gcurve(graph = gd, color = color.red)
vt = gcurve(graph = gd2, color = color.red)

"""
 3. 物體運動部分, 木塊到達地板邊緣時停止執行
"""
while(cube.pos.x <= L*0.5 - size*0.5):
    rate(1000)
    cube.pos.x += v*dt
    xt.plot(pos = (t, cube.pos.x))
```

```
    vt.plot(pos = (t, cube.v.x))
    t += dt

print("t = ", t)
```

如果將程式碼用 Python IDLE 的編輯器開啟，在預設狀態下看到的顏色應該會很類似上面的樣子，編輯器會自動將在 Python 中有特殊功能的保留字、引號內的文字、註解等用不同的顏色標示出來，這樣能便於閱讀程式碼。Python 有兩種註解方釋

1. 多行註解：於兩行 **"""** 之間的文字
2. 單行註解：於 **#** 之後到該行結束為止的文字

直譯器在執行程式碼時會忽略註解的部分，雖然註解對程式運作沒有幫助，但是對使用者而言非常重要，如果沒有寫註解，很有可能連作者本人在過了幾天之後也忘記自己在寫什麼，更不用說要讓其他使用者看懂程式碼，所以請務必養成寫註解的習慣。

我習慣在一開始先寫清楚程式的名稱、功能、日期及作者，接下在參數設定之前寫上

```
from vpython import *
```

這是 Python 引用函式庫的語法，在預設狀態下不會引用 vpython 這個做物理模擬用的函式庫，所以我們需要加上這行程式碼，意思是從 vpython 這個函式庫引用所有的函式。另外也可以寫成

```
importm vpython as 自訂名稱
```

假設自訂名稱為 vp ，則要引用 vpython 中的函式時需要寫 vp.[函式名稱]；如果只寫 `importm vpython`，則要引用 vpython 中的函式時需要寫 vpython.[函式名稱]。由於我們是以做動畫為主，建議採用第一種寫法，這樣在引用 vpython 中的函式時只需要寫函式名稱就可以了。

整個程式大約可以分成三個部分：

1. 參數設定
2. 畫面設定
3. 物體運動

參數設定

在參數設定部分，我習慣將一些在程式常中會不斷用到的數值指定到對應的變數中，並幫這些變數取一些容易看懂的名稱。雖然 Python 3.X 版已經支援 Unicode，可以使用中文作為變數名稱，但還是建議以**英文字母**、**數字**及**底線**作為變數名稱，其中**英文字母區分大小寫**，**變數名稱開頭不能為數字**，**不能使用系統保留字**。理論上可以按照自己的喜好命名變數，但為了便於理解變數的用途，最好**使用有意義的名稱**，例如將木塊大小取名為 size。

我在此定義的變數有 size、L、v、t、dt，用途都已經寫在該行的註解中。其中時間間隔 dt 的數值要視實際需求調整，這是因為 VPython 是利用數值方法計算物體的受力、加速度、速度、位移……等物理量，如果代入的時間長度太長，得到的數值會有較大的誤差；但如果代入的時間長度太短，整個模擬動畫執行的時間會拉長。目前設定的值為 0.01，對這個模擬動畫而言已經夠精準了。

畫面設定

我們會用到的函式為 **canvas**、**box**、**graph**、**gcurve**，以下分別說明這幾個函式的語法。

canvas是英文的帆布、畫布，在 VPython 中用來產生顯示動畫的畫面，目前是藉由瀏覽器來顯示，Google Chrome 或 FireFox 都可以，理論上 Windows Edge 應該也行，不過我完全不用 Windows Edge 所以就沒有測試。[1] 在 VPython 6 以及更早的版本，函數名稱是 **display**，執行時會另外開啟視窗。畫面中右方為 +x 軸方向，上方為 +y 軸方向，射出螢幕方向為 +z 軸方向。在這個程式中我將開啟的動畫視窗命名為 scene（英文，場景）。通常會調整的選項為：

1. title: 畫面的標題，顯示於畫面的左上角。
2. width: 畫面寬度（水平方向）。
3. height: 畫面高度（鉛直方向）。
4. x、y: 畫面左上角於瀏覽器視窗中顯示的位置，不過看起來 VPython 會依照已經存在的物件寬度、高度自動調整。
5. center: 代表觀察者所在位置。

6. background: 背景顏色，vector 括號中的數字依序為紅、綠、藍三原色的比例，範圍在 0 ~ 1 之間。另外也可以使用已經命名的常用顏色。[2]

box是英文的箱子、盒子，在 VPython 中用來產生長方體，在這個程式中木塊 cube 和地板 floor 都是藉由 box 產生的。[3] 通常會調整的選項為：

1. pos: 長方體中心的位置，數值為向量，vector(x, y, z)，vector 也可以簡化為 vec。
2. length、height、width 分別為 x、y、z 方向的長度，也可以簡化為 size = vector(x, y, z)。
3. color: 長方體的顏色。

graph是英文的圖，在 VPython 中用來產生繪圖視窗，在 VPython 6 以及更早的版本，函數名稱是 **gdisplay**。[4] ，在這個程式中我將兩個繪圖視窗分別命名為 gd 和 gd2，分別用來畫木塊的 x-t 圖和 v-t 圖。通常會調整的選項和 canvas 相似，title、width、height、x、y 的功能已經介紹過了，這畫用到較不同的選項為：

1. xtitle: x 軸名稱。
2. ytitle: y 軸名稱。

gcurve 在 VPython 中用來產生於繪圖視窗上畫出連續的曲線，在這個程式當中兩個曲線分別命名為 xt、vt，分別顯示於 gd 及 gd2 。通常會調整的選項為：

1. graph: 顯示於哪一個繪圖視窗。
2. color: 曲線的顏色。

另外還有只畫上數據點的 **gdots**，畫長條圖用的 **gvbars**，不過在這裡還沒有用到。

物體運動

利用一個 while 迴圈每隔一小段時間 dt 更新一次物體的狀態，由於我希望當木塊到達地板邊緣時程式停止運作，因此在 while 裡設定的條件為

```
cube.pos.x <= L*0.5 - size*0.5
```

接下來逐行說明程式碼的用途。

1. `rate(1000)` 是指每秒更新動畫1000次。

2. `cube.pos.x += v*dt` 用來更新木塊的位置，cube.pos.x 用來讀取 cube 位置的 x 座標，將讀取到的值加上速度 v 乘以一小段時間 dt ，再重新指定給 cube 位置的 x 座標。
3. `xt.plot(pos = (t, cube.pos.x))` 用來畫木塊的 x-t 圖，線段的橫軸位置為時間 t ，縱軸位置為木塊位置 cube.pos.x。
4. `vt.plot(pos = (t, cube.v.x))` 用來畫木塊的 v-t 圖，線段的橫軸位置為時間 t ，縱軸位置為木塊速度 cube.v.x。
5. `t += dt` 用來更新時間，將 t 加上 dt ，再重新指定給 t，寫法等同於 `t = t + dt`。

結語

雖然這個動畫的效果非常單純，甚至不需要動畫應該就能想像出物體運動的樣子。但也因為如此，很適合作為第一個動畫，只要動畫有任何不符合物理原理的地方都很容易看出來。之後會再把加速度、力量、角度……等更多的物理量加到動畫中。

VPython官方說明書

1. canvas: http://www.glowscript.org/docs/VPythonDocs/canvas.html
2. color: http://www.glowscript.org/docs/VPythonDocs/color.html
3. box: http://www.glowscript.org/docs/VPythonDocs/box.html
4. graph: http://www.glowscript.org/docs/VPythonDocs/graph.html