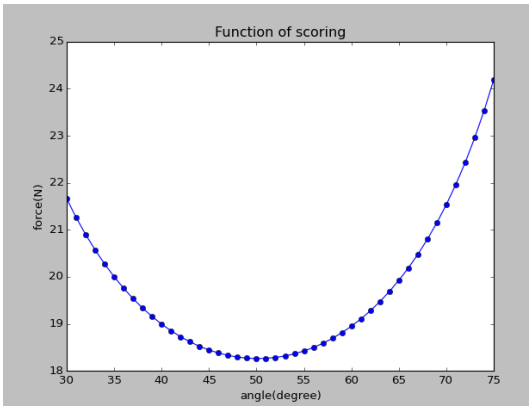
主題：我是神射手—探討投籃出手高度、角度、與力道的關係
組員：夏俊彥、鄭琮寶、石苯源

目的：
1.有空氣阻力的情況下，研究在三分線上出手時，根據不同的出手高度，能進球的角度與力道關係

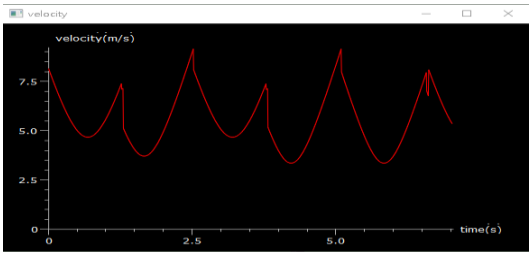
使用說明：

本程式會根據輸入的出手高度以及出手角度算出進球所需的力。當只輸入出手高度時，會顯示出手角度與所需的力的關係圖。除了”空心進球”，我們還考慮了彈框而進的因素，即籃球碰撞後的恢復係數。執行程式後，會跑出一個輸入介面如圖(1)，當輸入高度並按下提示後，會跑出出手角度與所需的力的關係圖如圖(2)；而當輸入高度以及出手角度後，於出手力道輸入欄位的後方會提示需要輸入多少力才會投進。注意高度不可低於球的半徑(0.2m)。可以根據提示在輸入端輸入所需的力，亦可參考提示在輸入端輸入提示值附近的力，觀察打板或彈框而進的情況。按下動畫會同時跑出”Velocity-time,速度與時間的關係圖”(圖(3))、”total energy,總能與時間的關係”(圖(4))以及”動能(綠線)及位能(紅線)與時間的關係圖”(圖(5))。

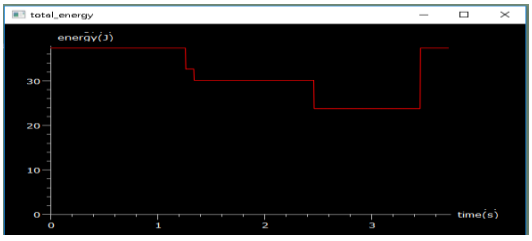
圖(1)



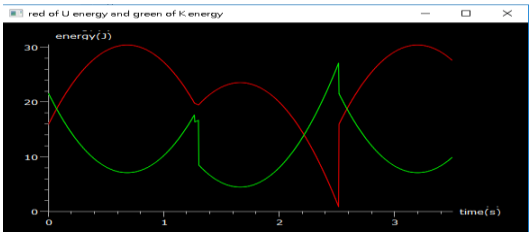
圖(2)



圖(3)



圖(4)



圖(5)

(一)、實驗所需參數

(1)籃框離地高度：3.0(m)

(2)籃框半徑：0.2(m)

(3)籃板寬：1.05(m)

(4)籃板長：1.8(m)

(5)籃球半徑：0.75/2π(m)

(6)場地寬：2(m)

(7)三分線距離：6.4(m)

(8)籃球重量：0.65(kg)

(9)籃框邊緣與籃板間距：0.15(m)

(10)恢復係數：0.85

(11)空氣阻力常數：0.47

(二)、模擬內容

1.空氣阻力

```
while (1==1):#動畫開始
    rate=100000
    dt=5*10**(-5)
    ball.velocity=ball.velocity-vector(0,9.8*dt,0)-((k*ball.velocity)*dt/m)#球之及時速度
    ball.pos=ball.pos+ball.velocity*dt#球之及時位置
    i=0
    j=0
    k=0
```

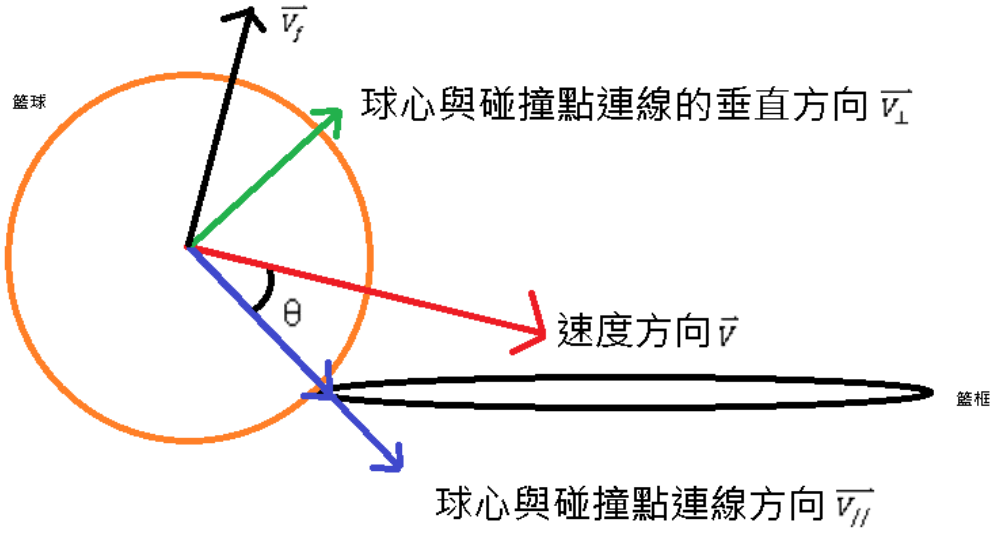
圖(1)

我們知道在雷諾數低的時候，空氣阻力大約和速度成正比，和速度方向相反，亦即$F_d = -kv$，其中k為阻力常數。因此我們在計算球的速度時，必須扣掉空氣阻力造成的速度衰減，如圖(1)

2.投籃彈框

```
if i==0:#碰撥籃框程式
    a=ball.velocity
    b=(basket.pos-vector(basket_radius,0,0))-ball.pos
    c=np.dot(a,b)*b/(abs(b))**2
    ball.velocity=ball.velocity-2*c*E
    i=1
ball.velocity=ball.velocity-vector(0,9.8*dt,0)-((k*ball.velocity)*dt/m)
ball.pos=ball.pos+ball.velocity*dt
```

圖(2)



圖(3)

在球與籃框碰撞的那一剎那，我們可以將球的速度方向$\vec{V}$分解為球心與碰撞點的連線方向$\overrightarrow{V_{//}}$以及與之垂直$\overrightarrow{V_\perp}$的方向如圖(3)。我們知道在與碰撞點的連線方向的速度碰撞後後來的速度分量會與原本差一個負號(假設彈性碰撞)即-$\overrightarrow{V_{//}}$，因此根據運動獨立性，我們可以將-$\overrightarrow{V_{//}}$與$\overrightarrow{V_\perp}$合成出碰撞後的速度$\overrightarrow{V_f}$。其中$\overrightarrow{V_f} = \overrightarrow{V_\perp} - \overrightarrow{V_{//}}$，又$\overrightarrow{V_\perp} = \vec{V} - \overrightarrow{V_{//}}$，因此我們可以得出$\overrightarrow{V_f}$與$\vec{V}$的關係式：$\overrightarrow{V_f} = \vec{V} - 2 \times \overrightarrow{V_{//}}$。$\overrightarrow{V_{//}}$又可利用正射影公式：$\frac{\vec{V}\cdot\overrightarrow{V_0}}{|\overrightarrow{V_0}^2|} \times \overrightarrow{V_0}$，$\overrightarrow{V_0}$為球心到籃框的方向向量。如圖(2)。a 為球的速度方向$\vec{V}$、b 為球心到籃框的方向向量$\overrightarrow{V_0}$、c 為利用正射影公式得到的$\overrightarrow{V_{//}}$。由於我們籃球有一恢復係數 0.85，因此我們得到碰撞後的速度乘上恢復係數後，模擬真實投籃彈框的情況。

3.二維拋體軌跡方程式

```
def f(x,h):#利用出手高度以及出手角度估算出手所需之速度
    global field_length,basket_radius,contact_length
    x=x*np.pi/180.0
    y=field_length-basket_radius-contact_length
    v=((9.8*y**2)/(2*(np.cos(x))**2)/(y*np.tan(x)-(3-h)))**0.5
    return v
```

圖(4)

我們知道在 2 維拋體的情況下，物體的軌跡方程式為：

$$y = x\tan\theta - \frac{gx^2}{2(v\cos\theta)^2} \quad (1)$$

給定出手高度以及角度後，利用已知籃框的位置，我們可以得到出手所需要的速度如圖(4)，進而換算成所需要的力(牛頓)。在有空氣阻力的情況下，仍利用此方程式計算的原因為：我們發現，由於速度不快，距離太短以及球面積不大，空氣阻力對整個系統能量的耗損並不大，此可以從總能與時間的關係圖看出。

4.防止卡牆或球黏在籃框上

```
while abs(ball.pos-(basket.pos-vector(basket_radius,0,0)))<basketball_radius:#碰卡牆或卡框程式
    if i==0:#碰撞籃框程式
        a=ball.velocity
        b=(basket.pos-vector(basket_radius,0,0))-ball.pos
        c=np.dot(a,b)*b/(abs(b))**2
        ball.velocity=ball.velocity-2*c*E
        i=1
    ball.velocity=ball.velocity-vector(0,9.8*dt,0)-((k*ball.velocity)*dt/m)
    ball.pos=ball.pos+ball.velocity*dt
```

圖(5)

由於程式運行時，都是非連續的運算(宏觀似連續，微觀非連續)，所以在運行判斷碰撞牆壁時的判斷式，因為在碰到牆壁反彈時的第一次判斷後，程式只會進行一小段移動，使得第二次判斷以為球還在碰撞中，造成二次碰撞，使得球卡在牆壁上的情形出現，因此我們必須做一個修正，讓他在判斷後，還未離開碰撞範圍時，持續移動，再進行下一次的判斷，避免球體在牆壁上震動的情形產生如圖(5)。

5. 進球角度及提示

```
try:#判斷角度有內容物有機會進球、有內容物沒機會進球和無內容物
    ang=float(entry2.get())
    if (((f(ang,h)*np.sin(ang*np.pi/180.0))**2)/2.0/9.8)>High:
        a1=0
    else:
        a1=1
except:
    a1=2
```

圖(6)

```
elif a1==1:
    theta=float(entry2.get())
    s="%.2f"%(f(theta,h)*m/0.3)
    label5=tk.Label(window,text="無法進球",font=(None,15))
    label5.grid(row=5,column=4,columnspan=1)
```

圖(7)

```
if a1==0:
    theta=float(entry2.get())
    s="%.2f"%(f(theta,h)*m/0.3)
    label5=tk.Label(window,text="hint:"+str(s),font=(None,15))
    label5.grid(row=5,column=4,columnspan=1)

    a=np.linspace(30,75,46)
    b=[]
    for i in a:
        b.append(f(i,h)*m/0.3)

    plt.title("Function of scoring")
    plt.xlabel('angle(degree)')
    plt.ylabel('force(N)')
    plt.plot(a,b,"-o")
    plt.legend(loc=2)
    plt.show()
```

圖(8)

```
elif a1==2:

    label5=tk.Label(window,text="                                ",font=(None,15))
    label5.grid(row=5,column=4,columnspan=1)
    a=np.linspace(30,75,46)
    b=[]
    for i in a:
        b.append(f(i,h)*m/0.3)

    plt.title("Function of scoring")
    plt.xlabel('angle(degree)')
    plt.ylabel('force(N)')
    plt.plot(a,b,"-o")
    plt.legend(loc=2)
    plt.show()
```

圖(9)

由於球有一個半徑存在，我們發現，當進球角度過於平緩時，無論如何都不可能進球，因此我們在計算需要多少力的時候，還必須判斷當球達到最高點時，球最低點的高度必須高於籃框。當最低點比籃框還低時，亦即出手仰角不夠時，在提示的地方就會顯示無法進球如圖(7)。若給定高度及角度，便會根據輸入的高度及角度提示投進所需的力如圖(8)。若只給高度則只會跑出角度與力的關係圖，如圖(9)

(三)結論

無論投籃高度如何，我們都可以從圖(2)的力與角度的關係圖看出，在曲線的最低點，差不多的力可以對應到一個範圍比較大的角度，因此在曲線的底部是投籃命中率最高的力道。從圖(4)我們也可以看出，球還在空中飛行(未碰撞)的情形下，空氣阻力對總能的影響趨近於0，這也是為什麼我們在計算時可以直接利用理想的軌跡方程式。

(四)參考資料

1.
https://zh.wikipedia.org/wiki/%E9%98%BB%E5%8A%9B#%E9%9B%B7%E8%AB%BE%E6%95%B8%E5%BE%88%E4%BD%8E%E6%99%82%E7%9A%84%E9%98%BB%E5%8A%9B