COMPUTACIÓN I – G.516-5

PROYECTO FINAL

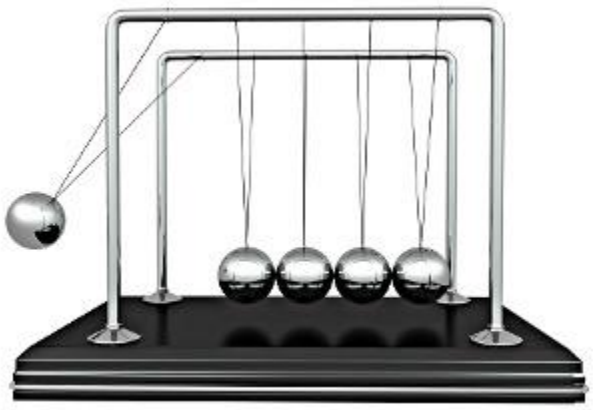
PENDULO/CUNA DE NEWTON

Por: Ignacio Ramos Galindo

1. INTR*ODUCCIÓN:*

El péndulo o cuna de Newton es un dispositivo mediante el cual se demuestran los teoremas de la conservación de la energía y del momento lineal o cantidad de movimiento.

El dispositivo cuenta con varios péndulos idénticos, alineados horizontalmente y prácticamente en contacto cuando están en reposo. En el modelo real cada una de las esferas está suspendida de dos hilos de igual longitud inclinados el mismo ángulo en sentido contrario (simulando una cuna). Esto es para restringir el movimiento al mismo plano vertical.



Su funcionamiento es simple: si desplazamos alguna de las esferas un cierto ángulo y las dejamos caer, colisionarán con las esferas que inicialmente están en reposo transfiriéndoles su energía y provocando que las últimas bolas de la cadena realicen la misma trayectoria que realizaron las primeras pero en sentido contrario.

En el modelo construido de manera real influyen muchos factores, entre ellos el más importante es las pérdidas de energía en las colisiones (choques inelásticos). Debido a la dificultad para simular este tipo de choque, nosotros asumiremos que estas pérdidas no existen (choques elásticos) y solo se perderá energía por el rozamiento del aire (prácticamente despreciable).

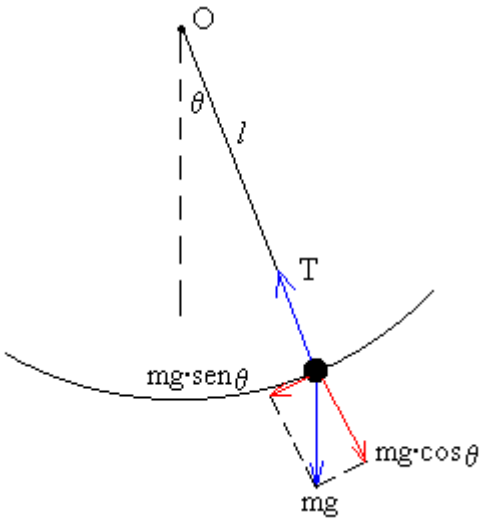
El objetivo principal de nuestro proyecto consistirá en realizar una simulación gráfica en tres dimensiones de este proceso físico, resolviendo previamente y numéricamente las ecuaciones implicadas en el proceso.

Una vez terminado el programa dispondremos de una herramienta que nos permitirá simular distintas situaciones, podremos modificar las condiciones iniciales tales como la masa de cada esfera, velocidades, ángulo iniciales, distancias… permitiéndonos jugar y recrear infinitas soluciones de las ecuaciones diferenciales.

2. REALICACIÓN DEL PROGRAMA:

2.1. Solución de la ecuación diferencial de un péndulo simple:

En primer lugar lo primero que debemos de hacer es plantear la ecuación que queremos resolver. Para ello nos basamos en el diagrama de fuerzas que tenemos a la derecha.



Si aplicamos la Segunda ley de Newton ($\sum F = m \times a$) podemos plantear la siguiente ecuación diferencial de segundo orden:

$$\frac{d^2\theta}{dt^2} + \frac{g}{L}\sin\theta = 0$$

Para hallar la solución a esta ecuación utilizaremos el método del punto medio, ya que su precisión es mucho mayor en comparación con el método de Euler.

Como es una ecuación diferencial de segundo orden tendremos que ir resolviendo simultáneamente la ecuación del ángulo y de la velocidad angular:

$$\omega(t + \Delta t) = \omega(t) - \frac{g}{L}\sin(\theta_{pm})\Delta t$$

$$\theta(t + \Delta t) = \theta(t) + \omega_{pm}\Delta t$$

Y el cálculo de los puntos medios ($\theta_{pm}\ y\ \omega_{pm}$) lo realizaremos por el método de Euler:

$$\omega_{pm} = \omega(t) - \frac{g}{L}\sin(\theta(t))\frac{\Delta t}{2}$$

$$\theta_{pm} = \theta(t) + \omega(t)\frac{\Delta t}{2}$$

Si queremos tener en cuenta también el rozamiento de la esferas con el aire, debemos introducir en el cálculo de $\omega(t+\Delta t)$ una componente $b\omega(t)$ y en el cálculo de ω_{pm} la componente $b\omega_{pm}$ en ambos casos con signo positivo, donde b es la constante de rozamiento del aire.

Para realizar estos cálculos en Matlab, en primer lugar definiremos las condiciones iniciales como las primeras componentes de un vector, y posteriormente resolveremos las ecuaciones planteadas por el método del punto medio mediante un bucle for desde un tiempo t = 0 hasta un tiempo final que le indiquemos:

```
%Inputs:
gra0 = input('Ángulo inicial en grados(Ten en cuenta el signo): ');
rad0 = (pi*gra0)/180;
w0 = input('Introducir velocidad inicial en rad/s: ');
m = input('Introducir la masa de la esfera: ');
L = 10;
R = 2;
g = 9.81;
dt = 0.1;
N = 10000; % Tiempo final.
u = 1.56e-5; % Viscosidad cinemática del aire a 25 grados Celsius.
b = (6*pi*u*R)/m; % Constante de rozamiento segun la ley de Stokes.
% Condiciones iniciales:
rad(1) = rad0;
w(1) = w0;
t(1) = 0;
x(1) = L*(sin(rad(1)));
y(1) = L-(L*(cos(rad(1))));
```

```
% Resolución por el método del punto medio.
for i = 1:N
    t(i+1)=t(i)+dt;
    % Calculo del Punto medio por el metodo de Euler:
    wpm = w(i)-((g/L)*sin(rad(i))+b*w(i))*dt/2;
    radpm = rad(i)+ w(i)*dt/2;
    % Solución de la ecuación:
    rad(i+1) = rad(i) + (wpm*dt);
    w(i+1) = w(i) - ((g/L)*sin(radpm)+b*wpm)*dt;
    % Posición de la esfera:
    x(i+1) = L*(sin(rad(i)));
    y(i+1) = L-(L*(cos(rad(i))));
end
```

La creación de los vectores **x** e **y** nos sirve para hallar la posición de la esfera a partir del ángulo obtenido al resolver la ecuación diferencial, y es necesario crearlos para poder realizar la animación posteriormente.

Por otro lado, como se puede observar, la **constante de rozamiento b** la obtenemos en función de unos parámetros concretos. Esto no es más que la "Ley de Stokes".

Esta ley nos indica que la constante de rozamiento para partículas esféricas en movimiento es:

$b = \frac{6\pi R\mu}{m}$ donde R es el radio de la esfera, m su masa y μ es la viscosidad del fluido (En nuestro caso hemos tomado la viscosidad cinemática del aire a 25ºC que es de aproximadamente 1.56e-5)

En nuestro caso no tenemos un solo péndulo, sino que hemos colocado cinco. Para ello simplemente dentro del bucle resolvemos cada vez estas ecuaciones diferenciales cinco veces, con la única diferencia de que cada una de las bolas tendrá determinada una pequeña separación respecto a la primera a la hora de definir su posición.

2.2. Determinar cuándo dos esferas colisionan:

Hasta ahora, solo hemos resuelto de forma simultánea las ecuaciones diferenciales de cinco péndulos simples. Pero la clave de este proyecto está en lo que ocurre cuando estos péndulos colisionan con sus adyacentes.

Por eso debemos determinar primero cuales son las condiciones que nos indican que dos bolas están en contacto para posteriormente poder aplicar las ecuaciones pertinentes.

Para ello en primer lugar, dentro del bucle he definido las distancias entre las bolas dos a dos (La primera con la segunda, la segunda con la tercera…) calculando el módulo del vector correspondiente:

```
% Distancias entre puntos:

d12(i) = ((((x2(i))-(x1(i)))^2)+(((y2(i))-(y1(i)))^2))^0.5;
d23(i) = ((((x3(i))-(x2(i)))^2)+(((y3(i))-(y2(i)))^2))^0.5;
d34(i) = ((((x4(i))-(x3(i)))^2)+(((y4(i))-(y3(i)))^2))^0.5;
d45(i) = ((((x5(i))-(x4(i)))^2)+(((y5(i))-(y4(i)))^2))^0.5;
```

Una vez calculadas las distancias, diremos que dos bolas están en contacto cuando la distancia entre ambas es igual a 2R.

Debido que estamos resolviendo un problema físico por aproximaciones, es posible que si el Δt introducido no es lo suficientemente pequeño las esferas se solapen, provocando que el programa no realice lo que nosotros queremos.

Para solucionar esto, en los condicionales que nos indicarán si una bola está en contacto o no, le impondremos una condición más fuerte. Se tendrá que cumplir que en ese instante la distancia entre ambas sea igual(o menor) a 2R, pero además, que en el instante anterior no sea igual(o menor) a 2R.

Si no pusiéramos esta condición, es posible que tras colisionar ambas esferas el programa realizara las operaciones de dentro del condicional varias veces, y nosotros solo queremos que las realice una vez.

```
if d12(i) <= (2*R) && d12(i-1) >= (2*R) % Choque 1

    ...

end


if d23(i) <= (2*R) && d23(i-1) >= (2*R)  % Choque 2.

   ...

end


if d34(i) <= (2*R) && d34(i-1) >= (2*R)  % Choque 3.

    ...
end


if d45(i) <= (2*R) && d45(i-1) >= (2*R)  % Choque 4.

    ...

end
```

.

2.3. Determinar que sucede cuándo dos esferas colisionan:

Cuando dos esferas están en contacto, para saber la dirección y el módulo de sus velocidades tras el choque, tendremos que imponer dos leyes.

Conservación de la Energía:

$$\frac{1}{2}m_1v_{1i}^2 + \cancel{m_1gh_1} + \frac{1}{2}m_2v_{2i}^2 + \cancel{m_2gh_2} = \frac{1}{2}m_1v_{1f}^2 + \cancel{m_1gh_1} + \frac{1}{2}m_2v_{2f}^2 + \cancel{m_2gh_2}$$

Y conservación del momento:

$$m_1v_{1i} + m_2v_{2i} = m_1v_{1f} + m_2v_{2f}$$

Dos ecuaciones con dos incógnitas (v_{1f} y v_{2f}) .

Sin embargo, resolver estas ecuaciones puede ser complicado, ya que puedes llegar a incongruencias como una ecuación de segundo grado que te indique dos soluciones y no obtener la opción más correcta.

Una solución alternativa consiste en cambiar el marco de referencia de manera que una de las velocidades conocidas es cero, resultando más fácil resolver las velocidades desconocidas de manera que si volvemos al marco original ya tenemos una de las velocidades que queríamos hallar y por simetría podemos determinar la que falta.

Las ecuaciones a aplicar son las siguientes:

$$v_1 = \frac{u_1(m_1 - m_2) + 2m_2u_2}{m_1 + m_2}$$

$$v_2 = \frac{u_2(m_2 - m_1) + 2m_1u_1}{m_1 + m_2}$$

Donde u1 y u2 son las velocidades de las respectivas esferas antes de la colisión y v1 y v2 las velocidades después de la colisión.

A la hora de determinar las velocidades iniciales, aunque no es del todo exacto, tendremos únicamente en cuenta la componente tangencial de la velocidad, ya que el error cometido no es muy grande y simplifica bastante el problema, de manera que los condicionales quedan de la siguiente forma:

```
if d12(i) <= (2*R) && d12(i-1) >= (2*R) % Choque 1

    % Velocidades justo antes de la colisión:

    u11 = (w1(i))*L;
    u21 = (w2(i))*L;

    % Calculo de las velocidades justo despues de la colisión:

    v11 = (((u11)*(m1-m2))+(2*m2*(u21)))/(m1+m2);
    v21 = (((u21)*(m2-m1))+(2*m1*(u11)))/(m1+m2);

    clear w1(i+1)
    clear w2(i+1)

    w1(i+1) = (v11)/L;
    w2(i+1) = (v21)/L;

end
```

Como se observa, en primer lugar se comprueba la condición de contacto. De ser así lo primero que hacemos es calcular las velocidades iniciales ($v = \omega \times R = \omega \times L$),

después aplicamos las formulas anteriores, y posteriormente volvemos a convertir las velocidades.

2.4. Dibujo 3D:

Para dibujar los resultados en tres dimensiones lo único que tenemos que hacer es crear dentro del bucle cinco esferas e ir modificando sus coordenadas de acuerdo a los parámetros x e y que se van obteniendo al resolver las ecuaciones diferenciales.

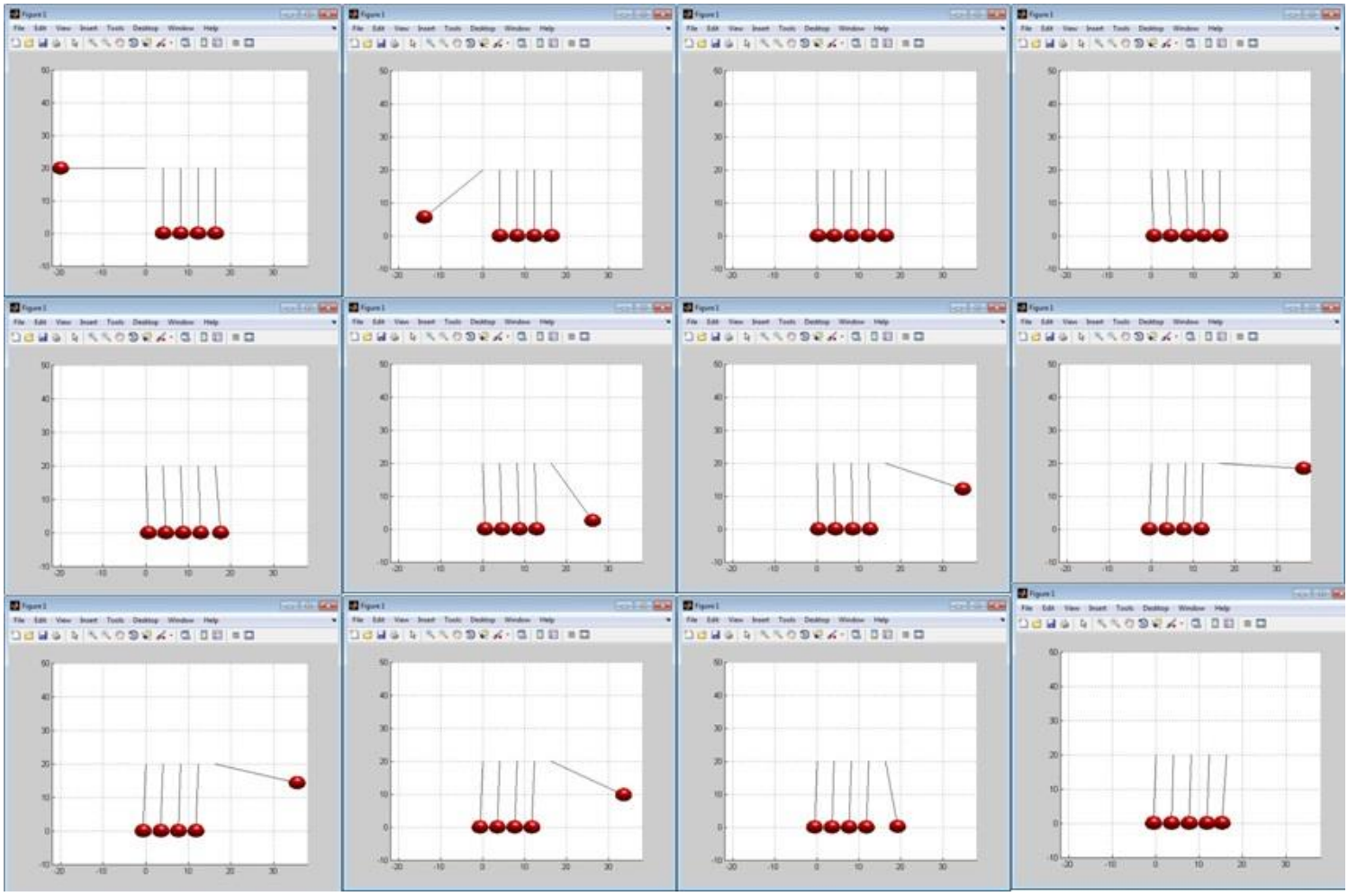
Vamos guardando cada una de las imágenes con el comando getframe, que posteriormente podremos reproducir con el comando movie:

```
%% Dibujo y simulación:
% Esferas:

[E1,F1,G1]=sphere();
surf(E1*R+x1(i), F1*R, G1*R+y1(i),'Facecolor','red','Edgecolor','none')
hold on
plot3 ([0 x1(i)], [0 0], [L y1(i)],'k-','LineWidth',1)
hold on

[E2,F2,G2]=sphere();
surf(E2*R+x2(i), F2*R, G2*R+y2(i),'Facecolor','red','Edgecolor','none')
hold on
plot3 ([(2*R+(d*R)) x2(i)], [0 0], [L y2(i)],'k-','LineWidth',1)
hold on

[E3,F3,G3]=sphere();
surf(E3*R+x3(i), F3*R, G3*R+y3(i),'Facecolor','red','Edgecolor','none')
hold on
plot3 ([(4*R+(2*d*R)) x3(i)], [0 0], [L y3(i)],'k-','LineWidth',1)
hold on

...
```

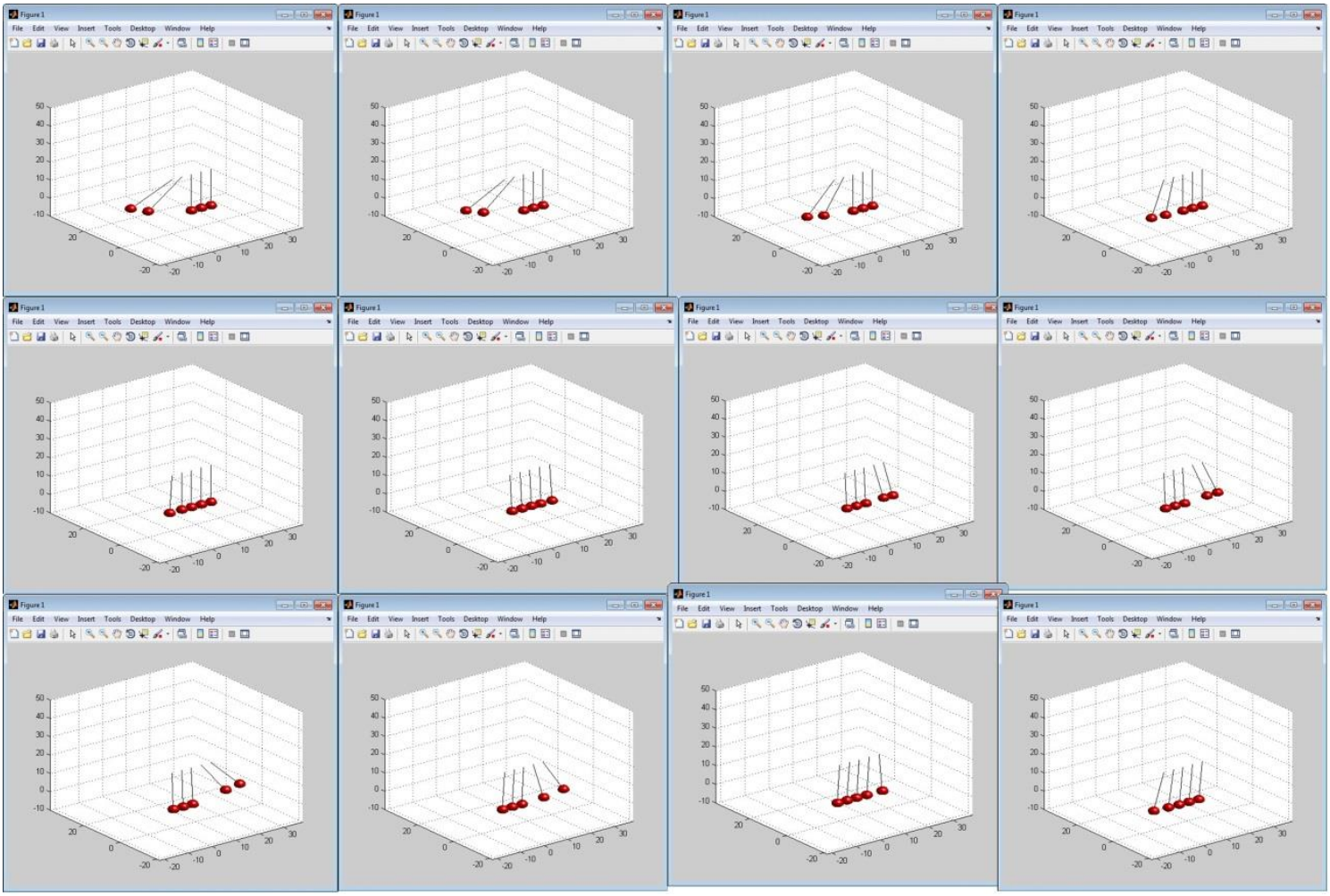
```
% Animación

axis([-1.5*L+(4*R) 1.5*L+(4*R) -1.5*L+(4*R) 1.5*L+(4*R) -0.5*L 2.5*L])
grid on;
camlight right;
lighting phong;
if VW == 1;
view([0 -90 0])
end
M(i)= getframe;
```

3. SIMULACIÓN DE ALGUNOS EJEMPLOS*:*

Lanzamiento de la esfera nº1 desde un ángulo de -90º sin velocidad. Masa de las esferas 10 kg:



Lanzamiento de la esfera nº1 desde un ángulo de -60º y de la esfera nº2 desde -45º sin velocidad. Masa de las esferas 10 kg



4.DISCURSIÓN DE LOS RESULTADOS:

Las simulaciones creadas a partir de distintas condiciones iniciales son bastante interesantes y la visualización en tres dimensiones da una buena impresión realista.

Bien es cierto que el experimento difiere bastante de la realidad. Estamos teniendo en cuenta choques en los que no se pierde energía, así como diversos parámetros tomados por aproximaciones. Aun así el programa nos permite hacernos una idea de lo que sucederá en distintas situaciones de forma automática y visual.

El programa presenta algunos inconvenientes.

Uno de ellos es que, para que el programa funcione es necesario que Δt sea lo suficientemente pequeño, y como consecuencia, debido a que la cantidad de cálculos y gráficos 3D que tiene que realizar el programa dentro del bucle es muy difícil evitar que la simulación sea bastante lenta.

Por otro lado, uno de los inputs del programa es la velocidad inicial de la esfera. Esto también presenta un problema ya que si aumentamos la velocidad, para que los choques se realicen correctamente tendríamos que disminuir Δt volviendo otra vez al primer problema.

Además otro problema que causa el introducir una velocidad inicial, es que el rozamiento que experimentan las esferas no es demasiado elevado, y si la velocidad es muy grande los péndulos pueden experimentar giros completos con sus consecuentes problemas (Las colisiones de las esferas solo están programadas con sus adyacentes, además, las cuerdas solo son una representación por lo que si se enredan entre ellas se observan situaciones extrañas).

5. BIBILIOGRAFÍA:

http://www.worldscientific.com/doi/pdf/10.1142/S0218202511005763

http://www.sc.ehu.es/sbweb/fisica/dinamica/con_mlineal/cuna_newton/cuna_newton.htm

http://www.uam.es/jose.ortega

http://en.wikipedia.org/wiki/Elastic_collision

http://forum.lawebdefisica.com

http://www.mathworks.es