Quiz 3: Colisiones de cuerpos rígidos

Guillermo Andree Oliva Mercado

6 de junio de 2014

1 Introducción

Uno de los problemas más utilizados en el mundo de las simulaciones ingenieriles y juegos es las colisiones de cuerpo rígido, pero es también, a manera de crítica, uno de los problemas de mecánica clásica menos tratados en los libros de pregrado.

2 Colisión de dos discos

Una aproximación inicial del problema puede encontrarse en el libro de Hauser. En él se encuentra el problema de colisión de dos discos rígidos de masa M que están sobre una superficie sin fricción, como dos discos de *hockey*.

El cuerpo 2 está en reposo inicialmente. Entonces, considerando las variables de las figuras, tenemos, utilizando conservación del moméntum lineal:

$$M\vec{u}_1 - M\vec{v}_1 = M\vec{v}_2$$

y queda que (haciendo prod. punto en ambos lados):

$$M^2 v_2^2 = M^2(u_1^2 + v_1^2 - 2u_1 v_1 \cos\theta_1)$$

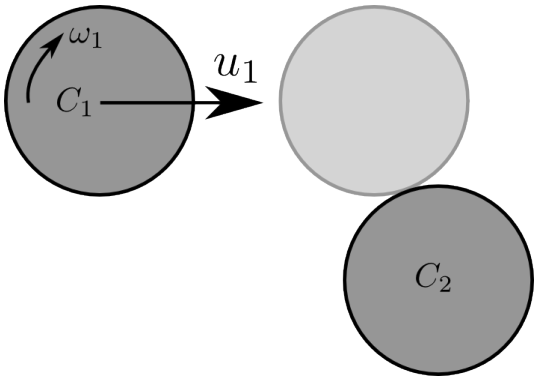

Figure 1: Discos: antes

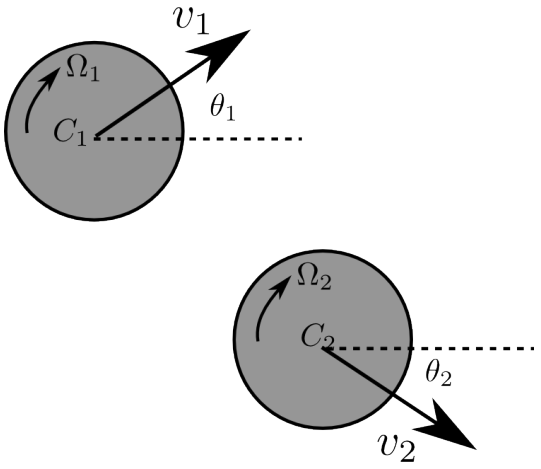

Figure 2: Discos: después.

y

$$\sin\theta_1 = \frac{v_2}{v_1}\sin\theta_2$$

Ahora definimos el ángulo α como el ángulo entre $\vec{u}_1$ y la normal de contacto, es decir, en este caso, la recta que une los centros de los dos discos. Aplicamos conservación del moméntum angular (todas en la dirección z).

- Antes del choque, $L_1 = I\omega_1 + Mu_1 R\sin\alpha; \quad L_2 = 0.$
- Después del choque, $J_1 = I\Omega_1 + MRv_1\sin(\alpha+\theta_1); \quad J_2 = I\Omega_2 + MRv_2\sin(\theta_2 - \alpha)$.

Por lo tanto, $L_1 = J_1 + J_2$, y pueden encontrarse $v_2, \quad v_1, \quad \Omega_2$ sabiendo las condiciones iniciales y Ω_1, o bien, si la colisión fuera también elástica, puede usarse conservación de la energía mecánica para saber también Ω_1.

En el caso de una simulación computacional, o también para determinar la trayectoria de todos los puntos, la colisión se produce cuando la distancia entre los dos centros de masa es igual a $2R$.

3 Cuerpo rígido (bloque) contra pared

Cuando hay cuerpos rígidos sin la simetría anterior, la situación se complica mucho. Ahora hay que buscar una manera de detectar la colisión, y otra manera de manejarla. El modelo que utilizaremos es el de *impulso* porque de cualquier manera discretizamos el tiempo, por lo que es posible determinar una fuerza y un torque que se aplicarán en un Δt dado. Las ecuaciones obtenidas son las mismas que la sección siguiente, con las dimensiones y masa infinitas de la pared.

La forma de detectar la colisión aquí será monitoreando los vértices de la caja, y cuando alguno haya penetrado la pared, se dispara el algoritmo de colisión.

3.1 Implementación del modelo con impulso de colisiones en Python.

Un bloque rectangular que va cayendo bajo la acción de gravedad constante, y se encuentra con un piso de masa y dimensiones infinitas (idealmente). En el formato siguiente, las líneas que comienzan con $>$ significa que son continuación de la línea anterior.

```
# coding=utf-8
```

importamos las librerías necesarias

```
from __future__ import division
from visual import *
```

construimos las cajas del piso y del cuerpo

```
c1 = box(pos=vector(0,2,0),height=1,width=0.01,axis=vector(0.6,0.4,0))
piso = box(width=0.01,height=0.1,length=8)
```

condiciones iniciales

```
c1.mass = 0.5 c1.vel = vector(0,0,0)
c1.angv = vector(0,0,0)
c1.ang = atan2(c1.axis.y,c1.axis.x)
c1.inertia = c1.mass*(c1.height**2+c1.width**2)/12
```

construimos las posiciones de los vértices

```
c1.a = c1.axis/mag(c1.axis)
# multiplication by [[0,-1][1,0]] a point transformation
c1.b = vector(-c1.a.y,c1.a.x,0)
c1.b = c1.b/mag(c1.b)
c1.vertices = [c1.length*c1.a/2+c1.b*c1.height/2,
>   -c1.length*c1.a/2-c1.height*c1.b/2, c1.height*c1.b/2-
>   c1.height*c1.a/2, -c1.height*c1.b/2+c1.a*c1.length/2]
```

valores iniciales del manejo de la colisión; def. del Δt

```
Collision = False
ind = None
Cr = 1
dt = 0.001
```

inicio del ciclo infinito de la simulación

```
    while True:
        rate(100)
```

detección de la colisión

```
        for i in range(4):
            if c1.pos.y + c1.vertices[i].y <= 0:
                Collision = True
                ind = i
                break
```

manejo de la colisión. Ver sección siguiente

```
        if Collision == True:
            normal = vector(0,1,0)
        rp1 = c1.vertices[ind]
            vp1 = c1.vel+rp1.cross(c1.angv)
            J = -(1+Cr)*vp1.dot(normal)/(1/c1.mass+mag2
    >           (rp1.cross(normal))/c1.inertia)
            Jv = J*normal
            c1.vel += Jv/c1.mass
            c1.angv += rp1.cross(Jv)/c1.inertia
            Collision = False
```

integración de las ecuaciones del movimiento

```
        c1.force = c1.mass*vector(0,-9.8,0)
        c1.acc = c1.force/c1.mass
        c1.vel += c1.acc*dt
        c1.pos += c1.vel*dt
        c1.ang += c1.angv.z*dt
        c1.axis = vector(cos(c1.ang),sin(c1.ang),0)*c1.length
        c1.a = c1.axis/mag(c1.axis)
        # multiplication by [[0,-1][1,0]] a point transformation
        c1.b = vector(-c1.a.y,c1.a.x,0)
        c1.b = c1.b/mag(c1.b)
        c1.vertices = [c1.length*c1.a/2+c1.b*c1.height/2,
    >      -c1.length*c1.a/2-c1.height*c1.b/2,
    >      c1.height*c1.b/2-c1.height*c1.a/2,
    >     -c1.height*c1.b/2+c1.a*c1.length/2]
```

El código tiene varias limitaciones debidas en gran parte al manejo numérico, por lo que necesita correcciones que son irrelevantes para el objetivo de este trabajo.

4 Colisión de dos cuerpos rígidos

Tal y como se describió en la sección anterior, la idea es aplicarle a los cuerpos un cambio de moméntum $\Delta\vec{p} = \vec{F}\Delta t = m\Delta\vec{v}$, en un tiempo Δt (en el código, `dt`).

El impulso será encontrado con

$$\vec{J} = m_1(\vec{v}_1^f - \vec{v}_1^f)$$

$$-\vec{J} = m_2(\vec{v}_2^f - \vec{v}_2^f)$$

que implica la conservación del moméntum lineal. Ahora, lo que nos interesa son las velocidades finales, por lo que

$$\vec{v}_1^f = \vec{v}_1^i + \vec{J}/m_1 \quad (1)$$

$$\vec{v}_2^f = \vec{v}_2^i - \vec{J}/m_2 \quad (2)$$

y también, utilizando el concepto de impulso para el torque, tenemos que

$$\vec{r}_{p1} \times \vec{J} = I_1(\vec{\omega}_1^f - \omega_1^i)$$

$$\vec{r}_{p2} \times \vec{J} = I_2(\vec{\omega}_2^f - \omega_2^i)$$

y de la misma manera,

$$\vec{\omega}_1^f = \vec{\omega}_1^i + \vec{r}_{p1} \times \vec{J}/I_1 \quad (3)$$

$$\vec{\omega}_2^f = \vec{\omega}_2^i + \vec{r}_{p2} \times \vec{J}/I_2 \quad (4)$$

Por otro lado, sea el punto P el que corresponde al punto de contacto entre los cuerpos. Las posiciones $\vec{r}_{p1}$y $\vec{r}_{p2}$ son las posiciones del punto P respecto al centro de masa de los cuerpos. Hacemos lo mismo con las velocidades, por lo que podemos definir

$$\vec{v}_{p1} = \vec{v}_1 + \vec{\omega}_1 \times \vec{r}_{p1} \quad (5)$$

$$\vec{v}_{p2} = \vec{v}_2 + \vec{\omega}_2 \times \vec{r}_{p2} \quad (6)$$

Además, definimos las velocidades relativas

$$\vec{v}_r^i = \vec{v}_{p1}^i - \vec{v}_{p2}^i \quad (7)$$

$$\vec{v}_r^f = \vec{v}_{p1}^f - \vec{v}_{p2}^f \quad (8)$$

Por último, es necesario definir un *coeficiente de restitución*, que no es más que, en cierta forma, una medida de cuánto se conserva o no la energía.

$$C_R = -\frac{\vec{v}_r^f \cdot \vec{n}}{\vec{v}_r^i \cdot \vec{n}} \quad (9)$$

donde $\vec{n}$ es el vector unitario normal en la dirección del contacto de los cuerpos.

Podemos definir $\vec{J} = J\vec{n}$, y resolver las ecuaciones anteriores para J. Sustituimos la ec. 8 en 7. Luego, introducimos las ec. 5 y 6. A la ecuación resultante

introducimos las ec. 3 y 4, y con eso, al simplificar, obtenemos la expresión para J siguiente:

$$J = -\frac{(1 + C_R)\vec{v}_r^{\,i} \cdot \vec{n}}{1/m_1 + 1/m_2 + (\vec{r}_{p1} \times \vec{n})^2/I_1 + (\vec{r}_{p2} \times \vec{n})^2/I_2}$$

con lo que después se pueden utilizar las ecuaciones 1-4 para obtener las velocidades lineales y angulares finales, y con ello integrar para obtener posiciones y desplazamientos angulares finales.

Reiteramos que para el caso de m_2, $I_2 \to \infty$, el J queda igual que el descrito en el código de la sección anterior.

References

[1] Ramtal, Dev; Dobre, Adrian (2011). *The Essential Guide to Physics for Flash Games, Animation, and Simulations.* ISBN 978-1-4302-3674-0

[2] Hauser, Walter (1969). *Introducción a los principios de mecánica.* UTEHA