Oblig 1: oppg 7.2 og 7.3
AST1100 – Høsten 2011

Aleksander Hansen

Oppg. 7.2

Problemet vi skal løse er et tilfelle av to-legeme problemet. Siden massen til Mars er mye mye større enn massen til Beagle 2 kan vi se bort i fra effekten Beagle 2 har på Mars og la sentrum av Mars ligge i massesentrumet til systemet. Vi bruker Newtons universelle gravitasjonslov $\vec{F}_g = -G\frac{m_1 m_2}{r^3}\vec{r}$ og Newtons 2. lov $\vec{F}_{res} = m\vec{a}$, samt en enkel modell for luftmotstand $\vec{F}_f = -k\vec{v}$ for å sette opp et uttrykk for bevegelsen til Beagle 2. Vi har at, $\vec{F}_{res} = \vec{F}_g + \vec{F}_f$ som gir oss diff. ligningen $m_1\ddot{\vec{r}} = -G\frac{m_1 m_2}{r^3}\vec{r} - k\dot{\vec{r}}$ hvor m_1 og m_2 henholdsvis er massen til Beagle 2 og Mars, G er Newtons gravitasjonskonstant, k er en friksjonskonstant og $\vec{r}$, $\dot{\vec{r}}$ og $\ddot{\vec{r}}$ er posisjonsvektoren og dens tidsderiverte.

Vi kan løse denne diff. ligningen numerisk med gitte initialbetingelser ved å bruke Euler-cromers metode. Det vi gjør da er å regne ut akselerasjonen i et punkt vha. Newtons 2. lov for så å finne hastigheten et tidssteg frem i tid og bruke den til å finne et nytt punkt et steg frem i tid. Deretter gjenntar vi prosessen til vi har oppnådd betingelsen for landing, altså at radiusen til banen til Beagle 2 er mindre eller lik radiusen til Mars.

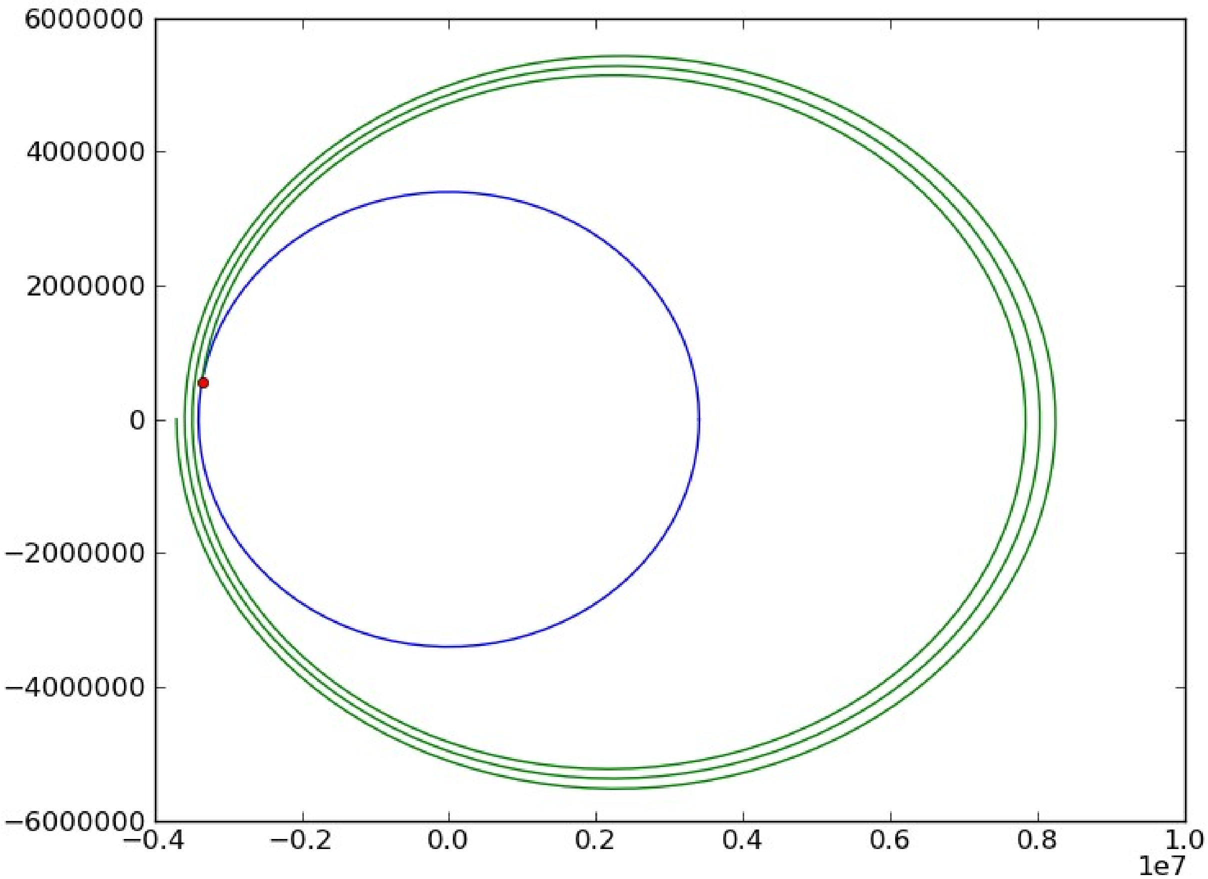
I koden under deklareres alle konstantene og initialbetingelsene først. To funksjoner, en for gravitasjon og en for friksjon er definert og disse blir kalt på i while-løkken under. Funksjonene regner ut henholdsvis gravitasjonskraften og friksjonskraften, og gir komponentene riktig tegn/retning. I while-løkken implementeres Euler-Cromers metode for å løse diff. ligningen som nevnt over. Banen til Beagle 2 plottes sammen med en sirkel som skal representere Mars og et rødt punkt som indikerer hvor Beagle 2 landet.

Python kode:

```
from scitools.all import *

# Constants:
M = 6.4e23    # Mass of Mars [kg]
m = 100.0     # Mass of Beagle 2 [kg]
G = 6.67e-11  # Newtons constant [N(m/kg)^2]
k = 1.6e-4    # Friction constant
R = 3.4e6     # Radius of Mars [m]
dt = 1.0      # Time step [s]

# Initial conditions for Beagle 2:
x0 = -3.698e6                 # x-coordinate[m]
y0 = 0.0                      # y-coordinate [m]
vx = 0.0                      # velocity component in the x-direction [m/s]
vy = -4.0e3                   # velocity component in the y-direction [m/s]
r = sqrt(x0**2 + y0**2)

# Gravitational force on m due to M at point (x,y):
def Gravity(x,y):
    r = sqrt(x**2 + y**2)
    theta = arccos(float(abs(x))/r)
    F = G*M*m/r**2
    if x<0:
        Fx = F*cos(theta)
    if x>0:
```

```
        Fx = -F*cos(theta)
    if x==0:
        Fx = 0
    if y<0:
        Fy = F*sin(theta)
    if y>0:
        Fy = -F*sin(theta)
    if y==0:
        Fy = 0
    return Fx, Fy, r

# Friction force on m at speed (vx,vy):
def Friction(vx,vy):
    v = sqrt(vx**2 + vy**2)
    phi = arccos(float(abs(vx))/r)
    f=k*v
    if vx<0:
        fx = f*cos(phi)
    if vx>0:
        fx = -f*cos(phi)
    if vx==0:
        fx = 0
    if vy<0:
        fy = f*sin(phi)
    if vy>0:
        fy = -f*sin(phi)
    if vy==0:
        fy = 0
    return fx, fy

# Declaring lists for x and y -coordinates
x = [x0]
y = [y0]

# Implementation of euler-cromers method
i = 0      # Loop variable
while r>=R:
    Fx, Fy, r = Gravity(x[i], y[i])
    fx, fy = Friction(vx, vy)
    ax = (Fx + fx)/m
    ay = (Fy + fy)/m
    vx = vx + ax*dt
    vy = vy + ay*dt
    x.append(x[i] + vx*dt)
    y.append(y[i] + vy*dt)
    #vx = vx + ax*dt            # If we instead update the velocities after we calculate the position,
    #vy = vy + ay*dt            # then we are using the ordinary euler method.
    i += 1

# Mars surface-coordinates
t = linspace(0,2*pi,360)
mx = zeros(360)
my = zeros(360)
for i in range(360):
    mx = R*cos(t)
    my = R*sin(t)

from point import *          # Just a simple module I made to plot just a single point
```

```
plot(mx,my)
hold('on')
plot(x,y)
point(x[-1],y[-1])
xlabel('x [m]')
ylabel('y [m]')
hardcopy('orbit.png')
```



Oppg 7.3

Jeg valgte å bruke Euler-Cromers metode istedet for bare Eulers metode fordi ellers går Beagle 2 over tre runder rundt Mars før den lander ~12° over ekvator med 1 sek. tidssteg. Med Euler-Cromers metode lander den på under 3 runder og mye nærmere ekvator som vi ser på bildet over. Dette stemmer bedre over med fasiten.