MATH-36031 Project2

ID:10636485

November 2022

1 Introduction

In this project, we will study two questions about a fox chasing a rabbit and simulate their positions at different times by solving differential equations. The initial configuration is shown in Figure 1. The fox is initially located[1] at (-250,-550) and the rabbit is at (0,0). The fox is chasing the rabbit with an initial speed $s_{f0} = 16m/s$ and the rabbit tries to hide from its predator and runs towards its burrow at (-600,600) in a straight line with initial speed $s_{r0} = 13m/s$. The fox's route to capturing the rabbit depends on whether the rabbit's view is blocked by an impenetrable warehouse, as follows:

1. If the rabbit is in sight, the fox's attack path will be directed towards the rabbit.

2. If the view of the rabbit is blocked by the corner S of an impenetrable warehouse, then the fox runs directly towards this corner. Once the fox has reached and gone past corner S, if the view of the rabbit is blocked or subsequently blocked, the fox will travel north parallel to the SN corner.

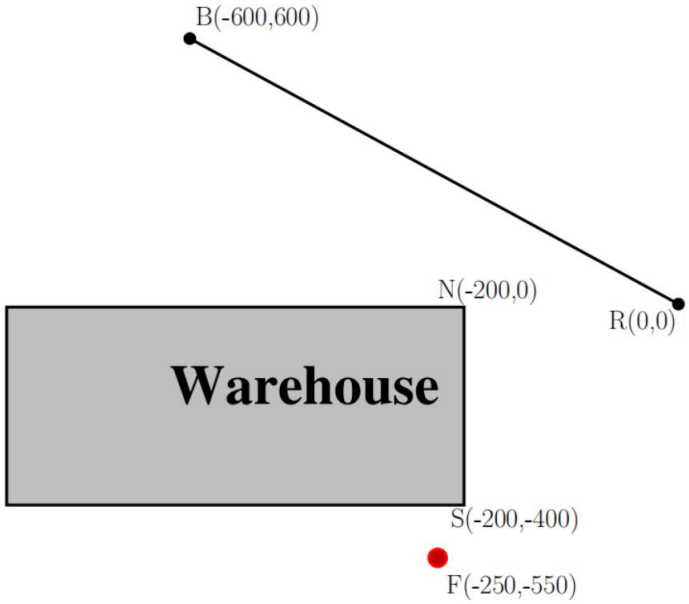

Figure 1: Coordinates (in metres) of the fox (F), the rabbit (R) and its burrow (B) and the two corners (S,N) of the warehouse.

2 Question 1: Constant speeds

Assuming that both the fox and the rabbit move with constant speeds $s_f = s_{f0} = 16m/s$ and $s_r = s_{r0} = 13m/s$ respectively, determine whether the rabbit can be captured before it reaches its burrow. If the distance between the rabbit and the fox is less than or equal to 0.1 meter, the rabbit is considered to be captured by the fox.

[1] The subscripts f and r denote the fox and rabbit respectively. The units of the coordinates are metres.

2.1 Problem solution

Our general idea for this problem is to set up the time step, where we can iterate through a while loop to calculate the coordinates of the fox and rabbit at each time step, then calculate the distance between them and determine whether the rabbit can be captured before it reaches its burrow.

Our first step is to find the position of the rabbit and the fox at time t. Since the trajectory of the rabbit is a straight line at an angle of 45 degrees to the horizontal, we can obtain

$$x_r(t) = -s_r t\sqrt{2}/2, \quad y_r(t) = s_r t\sqrt{2}/2$$

where x_r and y_r represent the horizontal and vertical coordinates respectively, t represents the time taken by the fox to chase the rabbit, s_r is the speed of the rabbit.

Then we define the function foxrab-ode to calculate the horizontal and vertical velocity of the fox. The horizontal and vertical velocity of the fox can be represented by

$$\frac{d}{dt}z_1(t) = s_f \frac{r_1(t) - z_1(t)}{\sqrt{(r_1(t) - z_1(t))^2 + (r_2(t) - z_2(t))^2}}, \quad \frac{d}{dt}z_2(t) = s_f \frac{r_2(t) - z_2(t)}{\sqrt{(r_1(t) - z_1(t))^2 + (r_2(t) - z_2(t))^2}}$$

We can obtain the coordinates of the fox (x_f, y_f) by solving the ODE. However, the trajectory of the fox depends on whether the rabbit's view is blocked by the warehouse. We can judge whether the view of the rabbit is blocked by determining whether the line segment SN intersects the straight line between the fox and the rabbit. The coordinates of S and N are (-200,-400) and (-200,0) respectively. If the line segment SN does not intersect the line fox-rabbit, it means that the rabbit is in sight. We have the following formula:

$$[(x_f - x_r)(-400 - y_f) - (y_f - y_r)(-200 - x_f)] \times [(x_f - x_r)(0 - y_f) - (y_f - y_r)(-200 - x_f)] \geq 0$$

If the view of the rabbit is blocked by the warehouse, it means that the line segment SN and the line fox-rabbit intersect, then we have:

$$[(x_f - x_r)(-400 - y_f) - (y_f - y_r)(-200 - x_f)] \times [(x_f - x_r)(0 - y_f) - (y_f - y_r)(-200 - x_f)] < 0$$

In this case, the fox will travel north parallel to the SN corner. The position of the fox can be expressed as:

$$x_f(t) = x_f(t), \quad y_f(t) = y_f(t) + s_f(\Delta t)$$

where $\Delta t = 0.001s$ is the time step.

Finally, the maximum distance between the rabbit and the burrow is $600\sqrt{2}$ metres and the distance travelled by the rabbit can be expressed as $s_r t$. If $s_r t \geq 600\sqrt{2}$, it means that the rabbit has reached the burrow. If the rabbit is captured by the fox, then the distance between them is less than or equal to 0.1 meter. The distance between the rabbit and the fox can be expressed as:

$$\sqrt{(x_f - x_r)^2 + (y_f - y_r)^2}$$

2.2 Code Implementation

```
s_r=13; % The constant speed of the rabbit
s_f=16; % The constant speed of the fox
x_r=0; % The initial position of the rabbit
y_r=0;
x_f=-250; % The initial position of the fox
y_f=-550;
x_r_all=x_r; % Store the initial position of the rabbit
y_r_all=y_r;
x_f_all=x_f; % Store the initial position of the fox
y_f_all=y_f;
d_r_max=600*sqrt(2); % Maximum distance between the rabbit and the burrow
d_r=0; % The initial distance travelled by the rabbit
d_f=0; % The initial distance travelled by the fox
d_fr = ((x_f-x_r)^2+(y_f-y_r)^2)^(1/2); % The initial distance between the rabbit and the
    fox
```

```
t=0; % The initial time
dt=0.001; % Time step
i=1; % The initial iteration
while (d_r < d_r_max) && (d_fr > 0.1) % The fox has not caught the rabbit and the rabbit has not reached the burrow
    t1=t; % Establish the tspan
    t2=t+dt;
    z0=[x_f,y_f]; % Initial condition
    [t_f,z]=ode45(@(t,z)foxrab_ode(t,z,s_r,s_f),[t1,t2],z0); % Solve the ODE to calculate the position of the fox and the time
    x_r=-s_r*(sqrt(2)/2) * t_f(end); % Update the position of the rabbit
    y_r=s_r*(sqrt(2)/2) * t_f(end);
    x_f=z(end,1); % Update the position of the fox
    y_f=z(end,2);
    d_r=s_r*t_f(end); % Update the distance travelled by the rabbit
    d_f=s_f*t_f(end); % Update the distance travelled by the fox
    SN=det_SN(x_f,y_f,x_r,y_r);
    d_fr = ((x_f-x_r)^2+(y_f-y_r)^2)^(1/2); % Update the distance between the rabbit and the fox
    x_r_all(i) = x_r; % Store the positions of the rabbit
    y_r_all(i) = y_r;
    x_f_all(i) = x_f; % Store the positions of the fox
    y_f_all(i) = y_f;
    i = i + 1; % Update the times of the iteration
    t=t+dt; % Update the time step
    while SN<0 % The view of the rabbit is blocked by the warehouse
        x_r=-s_r*(sqrt(2)/2) * t; % Update the position of the rabbit
        y_r=s_r*(sqrt(2)/2) * t;
        y_f=y_f+s_f*dt; % Update the position of the fox
        SN=det_SN(x_f,y_f,x_r,y_r);
        x_r_all(i) = x_r; % Store the positions of the rabbit
        y_r_all(i) = y_r;
        x_f_all(i) = x_f; % Store the positions of the fox
        y_f_all(i) = y_f;
        i = i + 1; % Update the times of the iteration
        t=t+dt; % Update the time step
    end
end

[x_f,y_f] % Output the final position of the fox
t % Output the total time
d_f % Output the distance travelled by the fox

plot(x_r_all,y_r_all,'LineWidth',1) % Plot the trajectory of the rabbit
hold on
plot(x_f_all,y_f_all,'LineWidth',1) % Plot the trajectory of the fox
line([-700,-200,-200,-700],[-400,-400,0,0],'color','k')
plot(x_f,y_f,'o','MarkerFaceColor','r') % Show the final position of the fox
legend('The trajectory of the rabbit','The trajectory of the fox')

function SN = det_SN(x_f,y_f,x_r,y_r)
% Determine whether the line segment SN and the line fox-rabbit intersect
term1 = ( x_f - x_r ) *( -400 - y_f ) -( y_f - y_r ) * ( -200 - x_f ) ;
term2 = (x_f - x_r)*(0 - y_f) - (y_f - y_r)*(-200-x_f);
SN = term1*term2;
end

function dzdt = foxrab_ode ( t ,z , s_r , s_f )
% Define the ODE function
% To calculate the horizontal and vertical velocity of the fox
r = [-s_r*sqrt(2)/2 * t, s_r*sqrt(2)/2 * t]; % the position of the rabbit
% the distance between the fox and the rabbit
dist = sqrt (( r (1) -z (1) ) ^2+( r (2) -z (2) ) ^2) ;
dzdt = zeros (2 ,1) ;% make sure the output is a column vector
dzdt (1) = s_f *( r (1) -z (1) )/ dist ; % horizontal velocity
```

```
77  dzdt (2) = s_f *( r (2) -z (2) )/ dist ; % vertical velocity
78  end
```

2.3 Result

The output is shown in Figure 2. We can easily find that the fox has not caught the rabbit. When the rabbit reached its burrow, the location of the fox was at (-448.40,410.74), the time it took was 65.27 seconds and the distance travelled by the fox was 1044.4 metres.

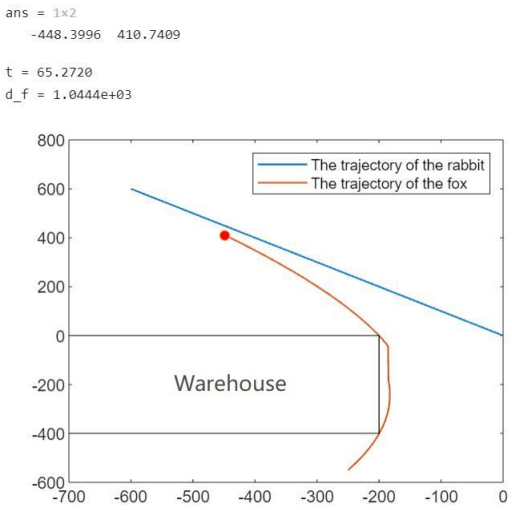

Figure 2: Command window output showing the location of the fox, the time, the distance travelled by the fox and the plot including the trajectory of the fox and the rabbit

3 Question 2: Diminishing speeds

Assuming that the speed of the fox's pursuit and the speed of the rabbit's escape diminish with time, according to the amount of distance they have travelled so far. More precisely, their speeds at time t are given by

$$s_f(t) = s_{f0}e^{-\mu_f d_f(t)}, \quad s_r(t) = s_{r0}e^{-\mu_r d_r(t)}$$

where $s_{f0} = 16m/s$ and $s_{r0} = 13m/s$ are the initial speeds, $\mu_f = 0.0002m^{-1}$ and $\mu_r = 0.0008m^{-1}$ are the rates of the diminishing speeds, $d_f(t)$ and $d_r(t)$ are the distance they have travelled up to time $t(> 0)$. Determine whether the rabbit can be captured before it reaches its burrow. (Suppose the diminishing speed starts from $t = 0$).

3.1 Problem solution

The general idea of this question is similar to the previous question but the way to find the location of the fox and the rabbit is different. Since the velocity is the rate at which displacement changes with time, we have

$$s_f(t) = d_f^{'}(t) = s_{f0}e^{-\mu_f d_f(t)}, \quad s_r(t) = d_r^{'}(t) = s_{r0}e^{-\mu_r d_r(t)}$$

We can model their positions at different times by solving ordinary differential equations. Then we have

$$d_r(t) = 1250log(\frac{13t}{1250} + 1), \quad d_f(t) = 5000log(\frac{2t}{625} + 1)$$

The position of the rabbit can be represented by:

$$x_r(t) = -1250log(\frac{13t}{1250} + 1)\sqrt{2}/2, \quad y_r(t) = 1250log(\frac{13t}{1250} + 1)\sqrt{2}/2$$

The position of the fox is shown as follows:
(1) If the rabbit is in sight, the position is (x_f, y_f) by solving the function foxrab-ode2.
(2) If the rabbit is not in sight, the position can be represented by:

$$x_f(t) = x_f(t), \quad y_f(t) = y_f(t) + s_{f0}e^{-\mu_f 5000log(\frac{2t}{625}+1)}(\Delta t)$$

3.2 Code Implementation

```
% Solve the ODE to calculate the distance travelled by the fox
syms y(t)
dsolve(diff(y)==16*exp(-0.0002*y),y(0)==0)

% Solve the ODE to calculate the distance travelled by the rabbit
syms y(t)
dsolve(diff(y)==13*exp(-0.0008*y),y(0)==0)

s_r0=13; % The initial speed of the rabbit
s_f0=16; % The initial speed of the fox
mu_r=0.0008; % The rates of the diminishing speeds
mu_f=0.0002;
x_r=0; % The initial position of the rabbit
y_r=0;
x_f=-250; % The initial position of the fox
y_f=-550;
x_r_all=x_r; % Store the initial position of the rabbit
y_r_all=y_r;
x_f_all=x_f; % Store the initial position of the fox
y_f_all=y_f;
d_r_max=600*sqrt(2); % Maximum distance between the rabbit and the burrow
d_r=0; % The initial distance travelled by the rabbit
d_f=0; % The initial distance travelled by the fox
d_fr = ((x_f-x_r)^2+(y_f-y_r)^2)^(1/2); % The initial distance between the rabbit and the
    fox
t=0; % The initial time
dt=0.001; % Time step
it=1; % The initial iteration
while (d_r < d_r_max) && (d_fr > 0.1) % The fox has not caught the rabbit and the rabbit
    has not reached the burrow
    t1=t; % Establish the tspan
    t2=t+dt;
    z0=[x_f,y_f]; % Initial condition
    x_r=-1250*log((13*t)/1250+1)*(sqrt(2)/2); % Update the position of the rabbit
    y_r=1250*log((13*t)/1250+1)*(sqrt(2)/2);
    [t_f,z]=ode45(@(t,z)foxrab_ode2(mu_f,t,z),[t1,t2],z0); % Solve the ODE to calculate the
        position of the fox and the time
    x_f=z(end,1); % Update the position of the fox
    y_f=z(end,2);
    d_r=1250*log((13*t)/1250+1); % Update the distance travelled by the rabbit
    d_f=5000*log((2*t)/625+1); % Update the distance travelled by the fox
    SN=det_SN(x_f,y_f,x_r,y_r);
    d_fr = ((x_f-x_r)^2+(y_f-y_r)^2)^(1/2); % Update the distance between the rabbit and the
        fox
    x_r_all(it) = x_r; % Store the positions of the rabbit
    y_r_all(it) = y_r;
    x_f_all(it) = x_f; % Store the positions of the fox
    y_f_all(it) = y_f;
    it = it + 1; % Update the times of the iteration
    t=t+dt; % Update the time step
        while SN<0 % The view of the rabbit is blocked by the warehouse
            y_f=y_f+16*exp(-mu_f*5000*log((2*t)/625+1))*dt; % Update the y-axis coordinates
                of the fox
            SN=det_SN(x_f,y_f,x_r,y_r);
            x_r_all(it) = x_r; % Store the positions of the rabbit
            y_r_all(it) = y_r;
            x_f_all(it) = x_f; % Store the positions of the fox
            y_f_all(it) = y_f;
            it = it + 1; % Update the times of the iteration
            t=t+dt; % Update the time step
        end
end

```

```
59 [x_f,y_f] % Output the final position of the fox
60 t % Output the total time
61 d_f % Output the distance travelled by the fox
62
63 plot(x_r_all,y_r_all,'LineWidth',1) % Plot the trajectory of the rabbit
64 hold on
65 plot(x_f_all,y_f_all,'LineWidth',1) % Plot the trajectory of the fox
66 line([-600,-200,-200,-600],[-400,-400,0,0],'color','k')
67 plot(x_f,y_f,'o','MarkerFaceColor','r') % Show the final position of the fox
68 legend('The trajectory of the rabbit','The trajectory of the fox')
69
70 function SN = det_SN(x_f,y_f,x_r,y_r)
71 % Determine whether the line segment SN and the line fox-rabbit intersect
72 term1 = ( x_f - x_r ) *( -400 - y_f ) -( y_f - y_r ) * ( -200 - x_f ) ;
73 term2 = (x_f - x_r)*(0 - y_f) - (y_f - y_r)*(-200-x_f);
74 SN = term1*term2;
75 end
76
77 function dzdt = foxrab_ode2 (mu_f, t ,z )
78 % Define the ODE function
79 % To calculate the horizontal and vertical velocity of the fox
80 r = [-1250*log((13*t)/1250+1)*(sqrt(2)/2), 1250*log((13*t)/1250+1)*(sqrt(2)/2)]; % the
      position of the rabbit
81 % the distance between the fox and the rabbit
82 dist = sqrt (( r (1) -z (1) ) ^2+( r (2) -z (2) ) ^2) ;
83 dzdt = zeros (2 ,1) ;% make sure the output is a column vector
84 dzdt (1) = 16*exp(-mu_f*5000*log((2*t)/625+1)) *( r (1) -z (1) )/ dist ; % horizontal
      velocity
85 dzdt (2) = 16*exp(-mu_f*5000*log((2*t)/625+1)) *( r (2) -z (2) )/ dist ; % vertical
      velocity
86 end
```

3.3 Result

The output is shown in Figure 3. We can see that the rabbit can be caught by the fox. The location of the fox was (-586.03,586.03), and the time $T = 90.46$ seconds. The distance travelled by the fox was 1271.2 metres.

ans =

$$5000\log\left(\frac{2t}{625}+1\right)$$

ans =

$$1250\log\left(\frac{13t}{1250}+1\right)$$

```
ans = 1×2
   -586.0317  586.0317

t = 90.4620
d_f = 1.2712e+03
```

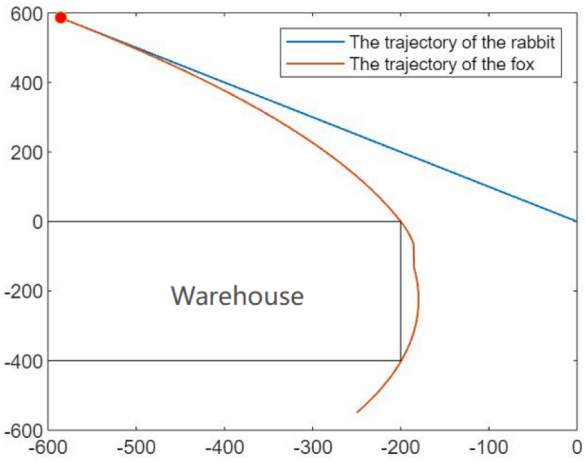

Figure 3: Command window output showing the solutions of ODE, the location of the fox, the time, the distance travelled by the fox and the plot including the trajectory of the fox and the rabbit