Problem Solving by Computer MATH36032 Coursework 2

This project explores the dynamics between a fox and a rabbit, by solving differential equations modelling their positions at different times. The initial configuration has the rabbit at $(0, 0)$, with a burrow at $(600, 600)$ and the fox at $(250, -500)$. There is a warehouse with corners $SW = (200, -400)$ and $NW = (200, 0)$, extending indefinitely to the east. The rabbit runs straight at its burrow, and the fox tries to run directly at the rabbit, with some conditions surrounding the warehouse that I will elaborate upon in my solutions.

Question 1: Constant speeds Assuming that both the fox and the rabbit run with constant speeds $s_{f_0} = 16m/s$ and $s_{r_0} = 13m/s$ respectively, determine whether the rabbit can be captured before it reaches its burrow.

I based my code heavily off of what was shown in the week 7 lectures in the form of the MATLAB live file and the functions `foxrab` and `foxode`. To create my solution, I started with the `foxode1` function.

I described the position of the rabbit at time t as a vector, using $v = d/t$, and multiplying each component by $\sqrt{2}/2$ to get the vector at 45 degrees to the x-axis, as that is along the straight line between the rabbits starting position and the burrow. Next I used an `if` statement to define the ODE based on which section of the journey the fox is in, and whether or not it can see the rabbit. The way I defined whether or not the fox could see the rabbit in the first section was by comparing the angle from the negative x-axis of the direction vector of the fox and the vector pointing from the fox to the corner SW. If the latter angle is smaller, the fox cannot see the rabbit. I used exactly the same idea for the second section, only with the corner NW and by comparing angles from the positive x-axis. Otherwise, the fox runs as described in the original `foxode` file.

```
function dzdt = foxode1(t, z, s_r, s_f)
% definition of the ODE

r = [sqrt(2)*s_r*t/2, sqrt(2)*s_r*t/2]; % the position of the rabbit
sw = [200,-400];
nw = [200,0]; % coordinates of corners of the warehouse

rdist = sqrt((r(1)-z(1))^2+(r(2)-z(2))^2); % the distance between the fox ...
    and the rabbit
swdist = sqrt((sw(1)-z(1))^2+(sw(2)-z(2))^2); % the distance between the fox ...
    and the corner SW

dzdt = zeros(2,1); % make sure the output is a column vector
dzdt(1) = s_f*(r(1)-z(1))/rdist; % horizontal velocity
dzdt(2) = s_f*(r(2)-z(2))/rdist; % vertical velocity

if( (z(2)<sw(2)) && ...
    (atan(abs(dzdt(2)/dzdt(1)))>=atan(abs((sw(2)-z(2))/(sw(1)-z(1))))) )
    % in the section below the warehouse, if rabbit is not visible, run ...
        straight towards SW
    dzdt(1) = s_f*(sw(1)-z(1))/swdist;
    dzdt(2) = s_f*(sw(2)-z(2))/swdist;
elseif( ((sw(2)<=z(2)) && (z(2)<nw(2)) ) && (atan(abs(dzdt(2)/dzdt(1)))<= ...
        atan(abs((nw(2)-z(2))/(nw(1)-z(1))))) )
    % in between SW and NW, if the rabbit is not visible, run north
    dzdt(1) = 0;
    dzdt(2) = s_f;
else
    dzdt(1) = s_f*(r(1)-z(1))/rdist;
    dzdt(2) = s_f*(r(2)-z(2))/rdist;
end
end
```

Next I wrote the events function, `foxrab1`. For my purposes, there are 2 events: the fox reaches the rabbit, and the rabbit reaches the burrow. Both the distance between the fox and rabbit, and the distance between the rabbit and the burrow, are decreasing, hence we have `direction = [-1, -1]`. If either event occurs, I want the ODE solver to stop (ie. I set the value of `isterminal` to 1).

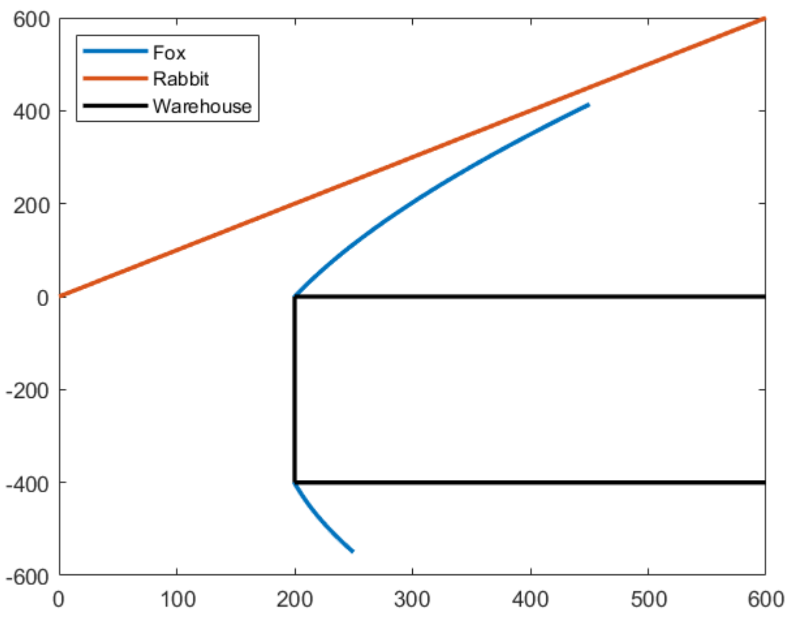
```
function [value, isterminal, direction] = foxrab1(t, z, s_r, mindist, burrow)
% events function for constant speeds

r = [sqrt(2)*s_r*t/2, sqrt(2)*s_r*t/2]; % the position of the rabbit

% first event: fox reaches (gets within minimum distance of) the rabbit
value(1) = sqrt((r(1)-z(1))^2+(r(2)-z(2))^2)-mindist; % distance between fox ...
    and rabbit
isterminal(1) = 1;
direction(1) = -1;

% second event: rabbit reaches burrow
value(2) = sqrt((r(1)-burrow(1))^2+(r(2)-burrow(2))^2)-mindist; % distance ...
    between rabbit and burrow
isterminal(2) = 1;
direction(2) = -1;
end
```

Finally I combine these two functions in a MATLAB live script, where I define the initial conditions, time span and minimum distance (0.1m in this case). I set the time span to be the time taken for the rabbit to run to the burrow, using $v = d/t$, with the speed of the rabbit and distance from it's starting point to the burrow. Then I solve the ode using the inbuilt `ode45` function, which takes care of the events with the `options` parameter. I have increased the accuracy of the solution with `'RelTol'` and `'AbsTol'`, as without these extra parameters I was having issues with the fox running through the warehouse. Lastly, I plot the movement of fox and rabbit, and I have also added the warehouse to the image for readability.

```
sr = 13;
sf = 16; % speeds of rabbit/fox
z0 = [250, -550]; % initial position of the fox
burrow = [600, 600]; % position of the burrow
ts = [0 norm(burrow)/sr]; % time span
mindist = 0.1;
options = odeset('RelTol',1e-8,'AbsTol',1e-10, ...
    'Events',@(t,z)foxrab1(t,z,sr,mindist,burrow));
[t,z,te,ze,zi]=ode45(@(t,z)foxode1(t,z,sr,sf),ts,z0,options);
te,ze,zi
plot(z(:,1),z(:,2),sqrt(2)/2*sr*t,sqrt(2)/2*sr*t, ...
    [200,600],[0,0],'k',[200,600],[-400,-400],'k', ...
    [200,200],[-400,0],'k','Linewidth',2)
lg = legend('Fox','Rabbit','Warehouse','Location','northwest');
```

```
te =
   65.2637
ze =
  450.3016  413.8831
zi =
     2
```

The outputs `te,ze,zi` tell us that event 2 was triggered (ie. the rabbit reached the burrow) after 65 seconds, when the fox was at position `ze`. Here's the plot of what happened:



As you can see, the fox runs around the warehouse perfectly, but unfortunately can't quite make it to the rabbit who reaches the burrow in the top right corner.

Question 2: Diminishing speeds The speeds of the fox and rabbit at time t are given by $s_f(t) = s_{f_0}e^{-\mu_f d_f(t)}$ and $s_r(t) = s_{r_0}e^{-\mu_r d_r(t)}$, where $s_{f_0} = 16m/s$ and $s_{r_0} = 13m/s$, $\mu_f = 0.0002m^{-1}$ and $\mu_r = 0.0008m^{-1}$ are the rates of their diminishing speeds and $d_f(t)$ 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.

For this question I used slightly altered versions of the `foxode1` and `foxrab1` functions, (namely `foxode2` and `foxrab2`) with included equations for the speeds of the rabbit and fox where necessary. At each time t the solver re-evaluates the values of the speeds using these equations, where I approximated the distance travelled using $v = d/t$. Apart from that, the basic ideas behind the structure of these functions did not change between questions.

```
function dzdt = foxode2(t, z, sr, sf)
% definition of the ODE including equations for the diminishing speeds

sf = sf*exp(-0.0002*sf*t); % equation for the diminishing speed of the fox
sr = sr*exp(-0.0008*sr*t); % equation for the diminishing speed of the rabbit
r = [sqrt(2)*sr*t/2, sqrt(2)*sr*t/2]; % the position of the rabbit
sw = [200,-400];
nw = [200,0]; % coordinates of corners of the warehouse

rdist = sqrt((r(1)-z(1))^2+(r(2)-z(2))^2); % the distance between the fox ...
    and the rabbit
swdist = sqrt((sw(1)-z(1))^2+(sw(2)-z(2))^2); % the distance between the fox ...
    and the corner sw

dzdt = zeros(2,1); % make sure the output is a column vector
dzdt(1) = sf*exp(-0.0002*sf*t)*(r(1)-z(1))/rdist; % horizontal velocity
dzdt(2) = sf*exp(-0.0002*sf*t)*(r(2)-z(2))/rdist; % vertical velocity

if( (z(2)<sw(2)) && ...
    (atan(abs(dzdt(2)/dzdt(1)))>=atan(abs((sw(2)-z(2))/(sw(1)-z(1))))) )
    % in the section below the warehouse, if rabbit is not visible, run ...
        straight towards SW
    dzdt(1) = sf*exp(-0.0002*sf*t)*(sw(1)-z(1))/swdist;
    dzdt(2) = sf*exp(-0.0002*sf*t)*(sw(2)-z(2))/swdist;
elseif( ((sw(2)<=z(2)) && (z(2)<nw(2))) && ...
    (atan(abs(dzdt(2)/dzdt(1)))<=atan(abs((nw(2)-z(2))/(nw(1)-z(1))))) )
    % in between SW and NW, if the rabbit is not visible, run north
    dzdt(1) = 0;
    dzdt(2) = sf*exp(-0.0002*sf*t);
else
    dzdt(1) = sf*exp(-0.0002*sf*t)*(r(1)-z(1))/rdist;
    dzdt(2) = sf*exp(-0.0002*sf*t)*(r(2)-z(2))/rdist;
end
end
```

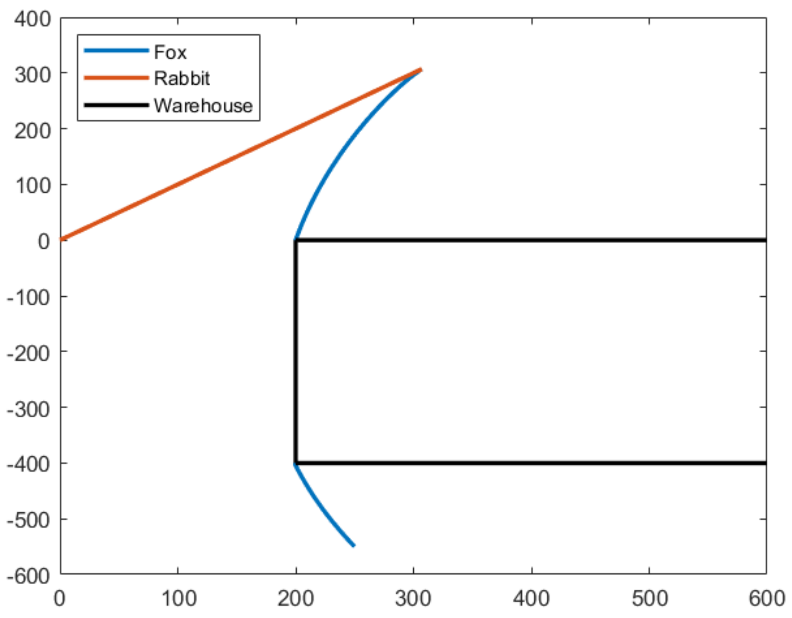
```
function [value, isterminal, direction] = foxrab2(t, z, sr, mindist, burrow)
% events function for diminishing speeds

sr = sr*exp(-0.0008*sr*t); % equation for the diminishing speed of the rabbit
r = [sqrt(2)*sr*t/2, sqrt(2)*sr*t/2]; % the position of the rabbit

% first event: fox reaches (gets within minimum distance of) the rabbit
value(1) = sqrt((r(1)-z(1))^2+(r(2)-z(2))^2)-mindist; % distance between fox ...
    and rabbit
isterminal(1) = 1;
direction(1) = -1;

% second event: rabbit reaches burrow
value(2) = sqrt((r(1)-burrow(1))^2+(r(2)-burrow(2))^2)-mindist; % distance ...
    between rabbit and burrow
isterminal(2) = 1;
direction(2) = -1;
end
```

Similarly, when I brought this all together in the MATLAB live file, it looked a lot like it did in the first question, but with two important changes. Firstly, I changed the time span to be the average of the time taken to run to the burrow at the initial and final speeds, as I feel this provides a good enough rough timeframe for the solutions.

```
sr0 = 13;
sf0 = 16; % initial speeds of rabbit/fox
z0 = [250, -550]; % initial position of the fox
burrow = [600, 600]; % position of the burrow
ts = [0, (norm(burrow)/sr0 + ...
    norm(burrow)/(sr0*exp(-0.0008*norm(burrow))))/2];
mindist = 0.1;
options = odeset('RelTol',1e-8,'AbsTol',1e-10, ...
    'Events',@(t,z)foxrab2(t,z,sr0,mindist,burrow));
[t,z,te,ze,zi]=ode45(@(t,z)foxode2(t,z,sr0,sf0),ts,z0,options);
te,ze,zi
plot(z(:,1),z(:,2),[0:ze(1)],[0:ze(2)], ...
    [200,600],[0,0],'k',[200,600],[-400,-400],'k', ...
    [200,200],[-400,0],'k','Linewidth',2)
lg = legend('Fox','Rabbit','Warehouse','Location','northwest');
```

```
te =
   67.2793
ze =
  307.1431  307.1311
zi =
     1
```

The outputs `te,ze,zi` tell us that event 1 was triggered (ie. the fox caught the rabbit) after 86 seconds, when the fox was at position `ze`. The second change I made was to the plot, as we now know that the rabbit runs straight to the point `ze` before being caught, so I drew its movement as the straight line up to the that point. Here's the plot of what happened:



Again the fox runs around the warehouse nicely, and manages to catch the rabbit soon thereafter. Given what we have seen in the first question, this outcome is to be expected since the fox is starting faster and has a lower rate of diminishing speed. It's also unsurprising, given its diminishing speed, to see that the rabbit has taken about as long to run half the distance as it took to run all the way to the burrow at constant speed.