Proceedings of The National Conference
On Undergraduate Research (NCUR) 2011
Ithaca College, Ithaca, New York
March 31-April 2, 2011

Pursuit Problems: Generalizations and Numerical Simulations

Greichaly Cabrera
Department of Mathematics
University of Puerto Rico at Humacao
Humacao, PR 00791-4300

Faculty Advisor: Pablo V. Negrón-Marrero

Abstract

A pursuit problem consists of studying the path followed by an aggressor (the pursuer) to catch a prey. This problem dates back to Zeno's solution of the classic *Achilles and the Tortoise problem,* Leonardo Da Vinci and Pierre Bouguer (1732). The term "pursuit curve" was introduced by George Boole in his *Treatise on differential equations* of 1859. The usual mathematical model in a pursuit problem is that of a differential equation that describes the relative velocity between the pursuer and prey and in which the speed of the pursuer is proportional to that of the prey. Besides the direct applications to biology, this problem is also important in ballistics and aviation.

In the classical pursuit problem, the prey follows a given known path and the problem is to determine the path followed by the pursuer. Normally the approach in textbooks and papers is to find exact solutions of the model equations which can only be obtained for fairly simple prey path curves. For more complex situations, numerical methods are the only practical alternative to approximate the solution.

A more realistic situation in a pursuit problem is that in which the prey follows a possibly random path. Another interesting situation is the one in which the pursuer chases more than one prey and has to make "decisions" on the fly onto which prey to follow. In this paper we propose models for both of these situations and perform numerical simulations to study the possible resulting trajectories followed by the pursuer.

1. Introduction

When a dog chases a cat or a rocket is sent to the moon, we are dealing with pursue problems. In the first case, the pursuer or the aggressor is the dog and the cat is the prey, while on the former, is the rocket that "chases" the moon. The pursuit problem consists of finding the path followed by the pursuer to catch the prey. This problem dates back to Zeno and Da Vinci, with the first mathematical treatment by Pierre Bouguer in his paper "Lines of Pursuit" (1732). In the classical model, the speed of the aggressor is proportional to that of the prey. Besides the direct applications of pursuit models to biology, this problem is also important in ballistics and aviation. The literature on the classical pursue problem is extensive. We refer to[1,2,4] for more detailed treatments and further references, and to[3] for other types of pursuit models called *motion camouflage*.

In this paper we study several generalizations of the classical pursue problem, with an emphasis on computer simulations. In particular, we consider the following three variations of the classical problem:

i) First we consider the situation in which the pursuer chases more than one prey and has to make "decisions" on the fly onto which prey to follow.
ii) Next we consider the case in which the proportionally constant between the speed of the pursuer and that of the prey, is not constant but a specified time dependent function. This could be thought as to be modeling the "fitness" of the pursuer.

iii) Finally we consider the case in which the prey follows a possibly random path or Brownian motion, also called a Wiener process. We include a mechanism into the model to make the path of the prey more variable or random as the pursuer gets closer.

The numerical simulations for models (i) and (ii) are done via standard ode's packages like those in MATLAB™. However these packages can not be used to solve stochastic differential equations (SDE) like in case (iii) above. Thus for the simulations in the third model we employ the SDE toolbox for MATLAB developed by U. Pucchini[5].

2. Classical pursuit model

In the classical pursuit problem, the aggressor moves directly toward the prey at each time *t*. We let $R(t) = (p(t), q(t))$ denote the position of the prey at time *t*, and by $F(t) = (x(t), y(t))$ that of the pursuer. For ease of exposition we are assuming the motion is in two dimensions but the discussion easily generalizes to higher dimensions. In the classical pursuit problem there are two basic assumptions:

i) The speed of the pursuer is proportional to that of the prey. This implies that whenever the prey runs faster, slower or stops, so does the aggressor.
ii) The velocity of the aggressor is always directed towards the position of the prey.

The first of these assumptions can be written as:

$$\|F'(t)\| = k\|R'(t)\|, \tag{1}$$

for some positive constant k. The second assumption above can be written as:

$$\frac{F'(t)}{\|F'(t)\|} = \frac{R(t)-F(t)}{\|R(t)-F(t)\|}. \tag{2}$$

If we combine equations (1) and (2), we get the basic equation relating the positions of the pursuer and the prey:

$$F'(t) = k\|R'(t)\|\frac{R(t)-F(t)}{\|R(t)-F(t)\|}. \tag{3}$$

Assuming the prey's path $R(t)$ is known, this differential equation together with some initial condition, determines the path $F(t)$ of the pursuer. In component form, Equation (3) can be written as:

$$x'(t)=k\frac{(p(t)-x(t))\sqrt{p'(t)^2+q'(t)^2}}{\sqrt{(p(t)-x(t))^2+(q(t)-y(t))^2}}, \quad y'(t)=k\frac{(q(t)-y(t))\sqrt{p'(t)^2+q'(t)^2}}{\sqrt{(p(t)-x(t))^2+(q(t)-y(t))^2}}. \tag{4}$$

We present two simulations for the classical pursuit problem using MATLAB's predefined function ode45 to solve equations (4). In one case (Figure 1) the prey follows a zigzagging path given by $p(t) = -\sin(t), \quad q(t) = t$, and in the other case (Figure 2) it follows a hyperbolic path given by $p(t) = -\cosh(t), \quad q(t) = \sinh(t)$. In the first case we set the constant *k* to 1.1, and for the hyperbolic path it was taken to be 1.5. The starting point for the pursuer in both cases was (5,0). We show in Figures 1 and 2 the results for both simulations. In each figure, the red curve represents the path followed by the prey while the blue one is that of the pursuer. In both cases the chase stopped when the distance between pursuer and prey was less than 0.001.

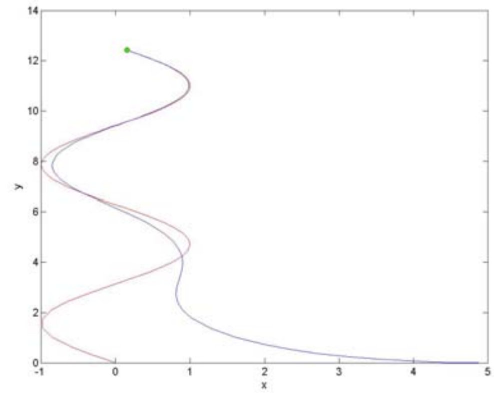

Figure 1. Simulation of a classical pursuit problem in which the prey follows a zigzagging path and k=1.1.

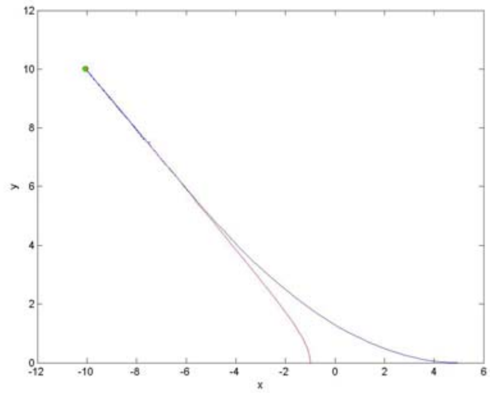

Figure 2. Simulation of a classical pursuit problem in which the prey follows a hyperbolic path and k=1.5.

3. Variations of the classical pursuit model

3.1 model of two preys

In the classical pursuit model we considered the simplest situation in which there is a single prey and a single aggressor. In real situations, like when one animal (the pursuer) is chasing a herd, the pursuer is chasing more than one prey and thus has to make "decisions" as to which prey to follow. For ease of exposition, we assume there are only two preys but the proposed decision mechanism can be applied to any number of preys. The criteria we propose to decide which prey will be followed is that of the minimum of the distances to the preys at any given time *t.* That is, the pursuer will follow whichever prey he determines to be closest. Thus we define the index $I(t)$ by:

$$I(t) = ind\left(\min\left\{\left\|R_1(t) - F(t)\right\|,\ \left\|R_2(t) - F(t)\right\|\right\}\right), \tag{5}$$

where $R_i(t) = (p_i(t), q_i(t))$ denotes the path of the i-th prey, $i = 1,2$. The index $I(t)$ is used to modify the classical equation (3) as follows:

$$F'(t) = k\left\|R'_{I(t)}\right\|\frac{\left(R_{I(t)}(t) - F(t)\right)}{\left\|R_{I(t)}(t) - F(t)\right\|}. \tag{6}$$

For the first simulation in this case we used similar looking zigzagging paths for both $R_1(t), R_2(t)$ given by:

$$R_1(t) = \left[-2 - \sin\left(0.7\left[t + \frac{\pi}{2}\right]\right), 0.7\left[t + \frac{\pi}{2}\right]\right], \quad R_2(t) = \left[3 - \sin(t), t\right].$$

Note that the path for R_2 which corresponds to the one on the right in Figure 3, starts closer to the pursuer but the prey corresponding to R_2 moves faster than the one on the left. With k=0.85 in the equations above and the pursuer starting at $(0,-4)$, we get that during the first part of the chase, the pursuer follows the prey on the right path, but then switches somewhere in the middle of the chase to the prey on the path to the left, because at that instant the prey on the left was closest. After that, the prey on path R_2 continues to get farther and the chase continues with the prey on the left until it is caught.

On the second simulation we have two preys moving in circular paths given by:

$$R_1(t) = [-1.1 + \cos(2t), \sin(2t)], \quad R_2(t) = [1.1 + \cos(t+\pi), \sin(t+\pi)],$$

which represent circles with center at (-1.1,0) and (1.1,0) respectively, with the prey corresponding to the path R_1 moving twice as fast as the other prey. Also note that $R_1(0) = (-0.1,0), R_2(0) = (0.1,0)$, which are the two points where the circles are closest to each other. With $k = 0.8$ is equation (6) , we show in Figure 4 the corresponding trajectories of the pursuer and prey. In this case we get, what appears to be from the numerics, a bi-stable orbit for the pursuer, as it chases each prey for sometime, and then switches back to the other, and so on. A video animation of this simulation can be downloaded from http://mate.uprh.edu/~pnm/vdo/dos_conejos.mp4.

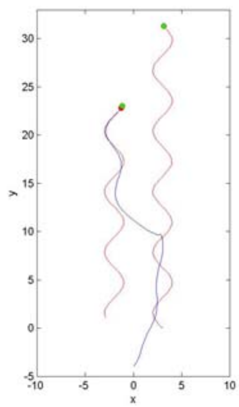

Figure 3. Chase of two prays illustrating the effect of the "decision" mechanism (5)

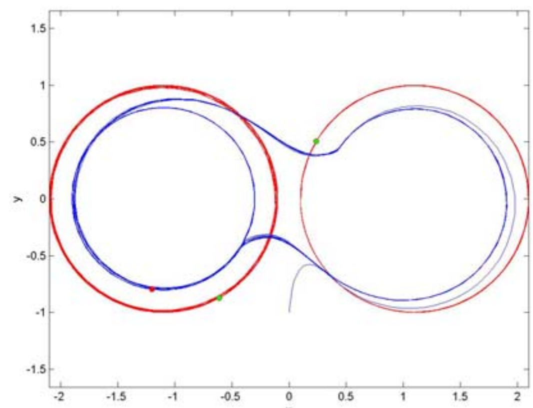

Figure 4: Chase of two preys in which the pursuer follows a possibly periodic trajectory.

3.2 model with a "fitness function" for the pursuer

This model gives us a more realistic view of the pursuit problem by including an additional component that gives more information about the pursuer. Thus we assume that the speed of the pursuer, instead of been proportional to that of the prey, it is given by a time dependent nonnegative function $G(t)$ times the speed of the prey. We call $G(t)$ the *fitness function* of the pursuer. It is reasonable to assume that $G(t)$ is a decreasing function of time so as to model that the pursuer gets tired as the chase progresses. In this model we modify the basic equation (3) as follows:

$$F'(t) = G(t)\|R'(t)\| \frac{R(t) - F(t)}{\|R(t) - F(t)\|}. \tag{7}$$

For the simulations we used the following function for $G(t)$:

$$G(t) = (k_1 - k_2)e^{-\alpha t} + k_2 \ , \tag{8}$$

where $k_1, k_2, \alpha > 0$ and $k_1 > k_2$. Here k_1, k_2 represent the initial and limiting fitness respectively. The parameter α controls how quickly is the transition from k_1 to k_2. In Figure 5 we show the resulting trajectories for the case in which $k_1 = 1.5$, $k_2 = 0.7$, $\alpha = 0.1$ and the prey follows the hyperbolic path $p(t) = -\cosh(t), \quad q(t) = \sinh(t)$. In this case the pursuer catches the prey to within an error of 0.05. In Figure 6 we show the same simulation but with $\alpha = 1.0$. In this case the pursuer gets "tired" rather quickly and can not catch the prey.

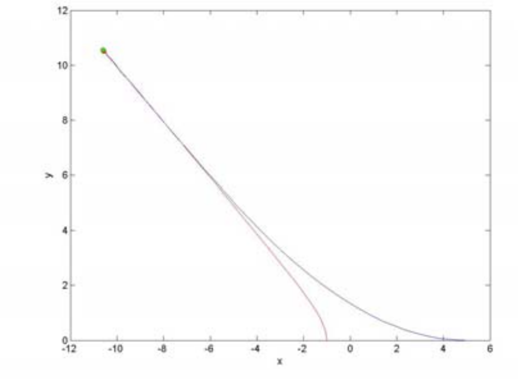

Figure 5. Simulation of a pursuit problem with fitness for the parameters $k_1 = 1.5$, $k_2 = 0.7$, $\alpha = 0.1$ in equation 8.

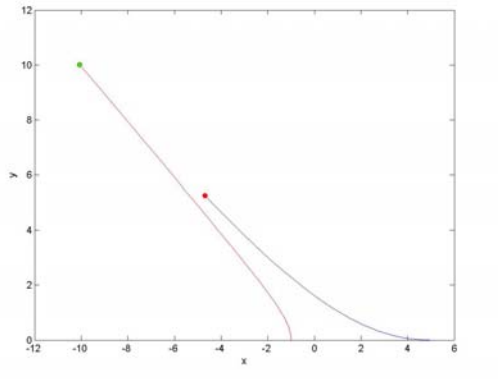

Figure 6. Simulation of a pursuit problem with fitness for the parameters $k_1 = 1.5$, $k_2 = 0.7$, $\alpha = 1.0$ in equation 8.

3.3 model using stochastic differential equations

We now consider the case in which the prey follows a possibly random path. In the pursuit models presented so far, the path of the prey (or preys) $R(t) = (p(t), q(t))$ is given or specified. Even though $R(t)$ is given, it can also be specified by a differential equation with an initial condition. Thus we specify the equation for the path of the prey as the solution of a differential equation of the form:

$$R'(t) = H(R(t)), \quad R(t_0) = R_0 . \tag{9}$$

To add random effects into the model, we consider for the path of the prey the *stochastic differential equation* (SDE):

$$R'(t) = H(R(t)) + B(R(t), F(t))W'(t), \quad R(t_0) = R_0 . \tag{10}$$

where $B(R(t), F(t))$ is a 2x2 matrix and $W(t)$ is a two dimensional Brownian motion or Wiener process. (We refer to Evans[6] for more definitions and theoretical aspects of SDE's.) The SDE pursuit model consists of the 4x4 system given by equations (3) and (10):

$$\begin{cases} R'(t) = H(R(t)) + B(R(t), F(t))W'(t), \\ F'(t) = k\|R'(t)\| \dfrac{R(t) - F(t)}{\|R(t) - F(t)\|} . \end{cases} \tag{11}$$

For the simulations we used the following form for the matrix $B(R(t), F(t))$:

$$B(R(t),F(t)) = e^{-\|R(t)-F(t)\|}\begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}. \quad (12)$$

This form for B makes the path of the prey more variable or random as the pursuer gets closer. We mention that standard ode's packages like those in MATLAB™ can not be used to solve stochastic differential equations like equation (11) above. Thus for the simulations we used the SDE toolbox for MATLAB developed by U. Pucchini[5] . In the toolbox's manual one finds more details of the numerical methods (Euler-Maruyama and Milstein) for SDE's. For our particular simulation, we took $R(t) = [\cos(t), \sin(t)]$ so that $H(R) = H(p,q)$ in (10) is given by $H(p,q) = (-q, p)$. Moreover, we take $k = 0.7$ in (11) and $\sigma_1 = \sigma_2 = 0.5$ in (12). With the prey starting at (1,0) and the pursuer at (5,0), we computed 20 trajectories for (11). In Figures 7 and 8 we show graphs for the components of each of the 20 trajectories computed for $R(t)$ and $F(t)$ respectively. We see from these graphs that they become more random like as time progresses and the pursuer gets close to the prey, due to the mechanism embodied in (12).

In Figure 9 we show in the left side the average trajectory of both the pursuer and prey, and on the right one of the generated trajectories. We can see that even though each single trajectory is highly oscillatory due to the random process used to generate it, the average trajectories resemble very much the trajectories of the corresponding deterministic problem ($\sigma_1 = \sigma_2 = 0.0$).

4. Conclusions

We discussed three variations or generalizations of the classical pursuit problem: the first involving two preys, a second one involving a "fitness function" for the pursuer, and a third model using a stochastic differential equation. Each model could be improved or make more realistic in many respects. For example, in the model with two preys, the criteria of the "closest" might not be the realistic one: the pursuer might consider instead the size of the prey, etc. In the model with fitness for the pursuer, one may consider different models for the fitness function and possibly a fitness function for the prey as well. Finally in the stochastic model one might experiment with different types of random processes and including random terms in the pursuer equation as well. Still the proposed models, although simple in their mechanisms, provide a good starting point for further generalizations of the classical or basic model.

5. Acknowledgements

This research has been funded in part by the NIH-RISE Program at the University of Puerto Rico at Humacao and by the NSF–PREM Program of the University of Puerto Rico at Humacao (Grant No. DMR–0934195). We thank Errol Montes-Pizarro of the University of Puerto Rico at Cayey who brought to our attention some of the interesting questions about pursuit problems considered in this paper in particular the situation with more than one prey.

6. References

1. Blanchard, P., Devaney, R. L., and Hall, G. R., *Differential Equations*, Brooks/Cole Publishing Company, 1-30, 1998.
2. Davis, H. T., *Introduction to Nonlinear Differential and Integral Equations*, Dover, New York, 1962.
3. Glendinning, P., "*The mathematics of motion camouflage*", Proceedings of the Royal Society of London, 2003.
4. Marshall, J. A., Broucke, M. E., and Francis, B. A., "*Pursuit formations of unicycles*", *Automatica* 42, 3-12, 2006.
5. Picchini, U., SDE toolbox: simulation and estimation of stochastic differential equations with MATLAB, http://sdetoolbox.sourceforge.net.
6. Evans, L. C., *An introduction to stochastic differential equations*, Lecture Notes, Department of Mathematics, UC Berkeley.

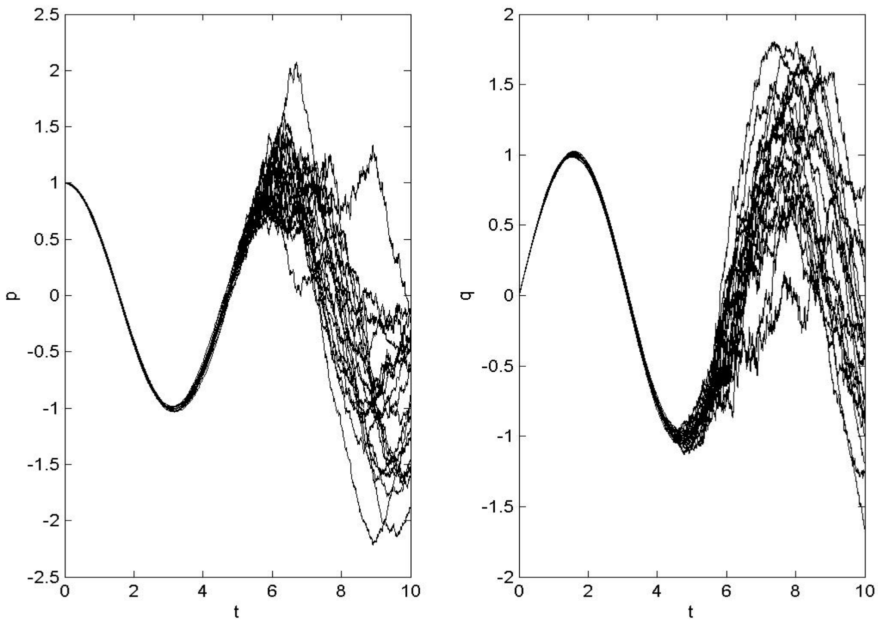

Figure 7. Components (p,q) for the trajectories generated for the model equation (11).

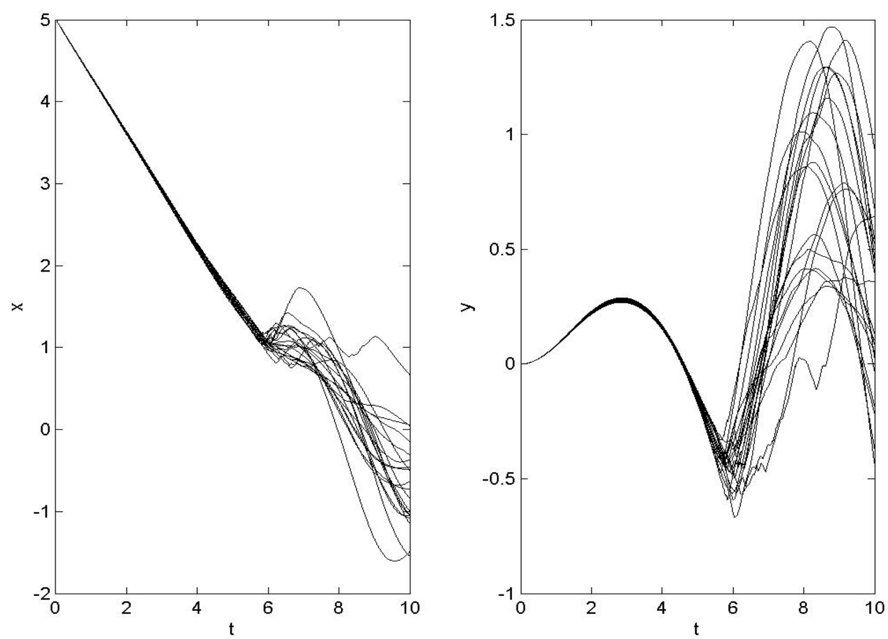

Figure 8. Components (x,y) for the trajectories generated for the model equation (11).

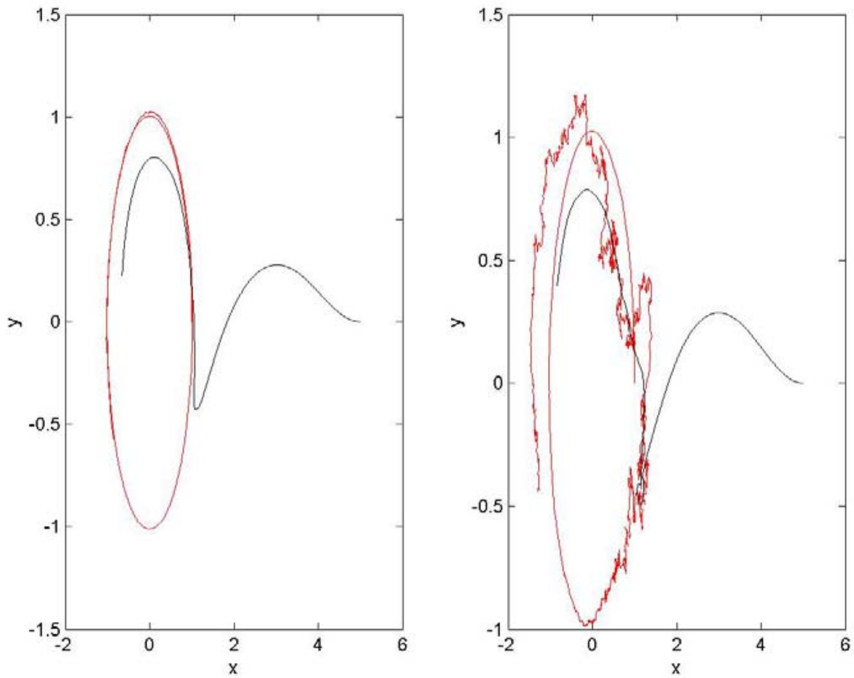

Figure 9. Average trajectory of both the pursuer and prey (left) and a particular instance of both trajectories (right) in the SDE model (11).