COOPERATIVE OPTIMAL PATH PLANNING FOR HERDING PROBLEMS

A Thesis

by

ZHENYU LU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2006

Major Subject: Aerospace Engineering

COOPERATIVE OPTIMAL PATH PLANNING FOR HERDING PROBLEMS

A Thesis

by

ZHENYU LU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	John E. Hurtado
Committee Members,	John Junkins
	Jianxin Zhou
Head of Department,	Helen Reed

December 2006

Major Subject: Aerospace Engineering

ABSTRACT

Cooperative Optimal Path Planning for Herding Problems. (December 2006)

Zhenyu Lu, B.Eng., Tongji University;

M.Eng., Texas A&M University

Chair of Advisory Committee: Dr. John E. Hurtado

In this thesis we study a new type of pursuit-evasion game, which we call the herding problem. Unlike typical pursuit evasion games where the pursuer aims to catch or intercept the evader, the goal of the pursuer in this game is to drive the evader to a certain location or region in the x-y plane. This herding model is proposed and represented using dynamic equations. The model is implemented in an effort to understand how two pursuers work cooperatively to drive multiple evaders to the desired destination following weighted time-optimal and effort-optimal control paths. Simulation of this herding problem is accomplished through dynamic programming by utilizing the SNOPT software in the MATLAB environment. The numerical solution gives us the optimal path for all agents and the corresponding controls as well as the relative distance and angle variables. The results show that the pursuers can work cooperatively to drive multiple evaders to the goal.

To my Lord and my family

ACKNOWLEDGMENTS

I am grateful to my advisor, Dr. John E. Hurtado, for all the time he has spent mentoring me. I would like also to extend my thanks to the other members of my committee, Dr. John Junkins and Dr. Jianxin Zhou, for their encouragement and support.

I also want to thank my family and friends for all of their support and their prayers and encouragement.

Finally, I have to give thanks to God for the blessings that He has given me. I know that all that I have comes from Him and pray that all that I do is for His glory.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

The traditional two-player pursuit-evasion contest is a well known problem. It has received serious attention from some areas, such as animal behavior [1] and differential game theory [2]. The optimal solutions of two-player pursuit and evasion games can be obtained by solving a mini-max optimal control problem [3]. The solution of lateral control acceleration is applied by the pursuer and this technique is known as Proportional Navigation Guidance (PNG). The PNG [4] [5] [6] method has been widely used in military applications such as aerial intercept guided missile systems [7]. In most of the pursuit evasion models considered to date, the pursuer's aim is to intercept or hunt the evader. Very natural and interesting extensions of the problems can be considered to cases where the interests of the pursuers and the evaders are not totally opposite [8]. Inspired by this extension, unlike the traditional pursuit and evasion games, the aim of the pursuer in our research is to drive the evader to a certain location or region in the x-y plane.

A dissertation by Shedied [9] formulated a basic herding problem as an optimal control problem. However, Shedied [9] only covered the two-player case. We extend the approach of Shedied [9] to consider the multi-player scenario and we suitably alter the mathematical model that governs the interaction between pursuers and evaders. Another type of herding problem is the study of shepherding behaviors [10] [11], in which one group (the shepherds) try to control the motion of another group (the flock). It mainly focuses on group behavior instead of the optimal path for the flock to travel to the goal.

The journal model is *IEEE Transactions on Automatic Control.*

Multiple robot systems have been a growing interest in many fields of research, such as search, rescue, delivery and reconnaissance. These systems can provide advantages over a single-robot system. The main goal of a team of robots with cooperation is to accomplish a certain task more efficiently than a single robot, or to complete a task that an equivalent single-robot cannot. For example, a thesis by Kang [12] presents cooperative multi-agent robot constellations for localization and entire path planning using a weighting function approach to modeling of irregular surfaces based on the finite element method. Feddema [13] describes how decentralized control theory can be used to analyze the control of multiple cooperative robotic vehicles with multirobot formation control, perimeter surveillance, and other applications. When used for localization of chemical sources, simulation and experiments have shown that by sharing concurrent sensory information, a group of robots can better estimate the shape of a chemical plume and, therefore, localize its source [14]. The source can even be time-dependent [15]. In summary, task complexity and environmental uncertainty suggest that solutions to complex problems can frequently benefit from cooperation between vehicles.

We report here research on multi-player cooperative herding problems, such as a system of two pursuers and multiple evaders. We will first formulate an optimal control problem and then focus on using dynamic programming methods to compute solutions numerically by utilizing the SNOPT software [16] in the MATLAB environment. SNOPT (invented by Philip Gill, Walter Murray and Michael Saunders), uses a sequential quadratic programming (SQP) algorithm, within a software package to solve large-scale linear and quadratic programming problems, linearly constrained optimization problems, as well as local approximation solutions to general nonlinear programming problems.

We introduce the simplest one pursuer and one evader system (1P1E scenario) first. The evader moves away from the pursuer according to two mathematical rules. First, the direction of the motion of the evader is modeled as along the straight line joining the pursuer and the evader. Secondly, the evader's velocity is bounded by the inverse of the distance between the evader and the pursuer. As the distance between the evader and pursuer decreases, the evader increases its velocity in the direction opposite to the pursuer. With these models, we formulate a basic herding problem as an optimal control problem. Once the 1P1E scenario is well established, a two pursuer and two or more evaders (e.g., 2P3E) scenario is considered. The motion of each evader is influenced by the relative vector of each evader. This can be thought of as an "influence" vector from each pursuer. If we want both pursuers to influence the motion of a single evader, we consider the summation of both influence vectors, one from each pursuer. With this cooperative model, we develop the formulation of optimal control problems and solve them numerically as done in the 1P1E scenario. The numerical solution gives us the optimal path for all agents and the corresponding controls as well as the relative distance and angle variables. The results show that two pursuers can work cooperatively to drive multiple evaders to the desired destination following weighted time-optimal and effort-optimal control paths.

CHAPTER II

ONE PURSUER AND ONE EVADER SIMPLE SYSTEM (1P1E)

For the simple system, we introduce two agents: a Pursuer (P), which represents a fully equipped robot with global navigation (e.g., GPS) and relative position sensors, and an Evader (E), which represents a relatively cheap robot with only relative position sensors. In the context of animal herding, the pursuer can be thought of as a shepherd and the evader as a member of a flock.

A. System Model

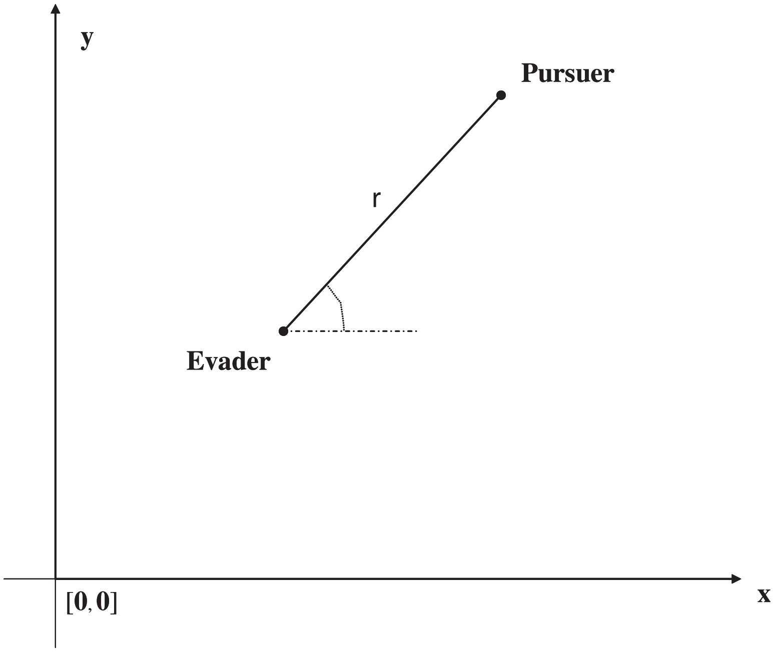

Fig. 1. System Model for 1P1E

Figure 1 gives us a representation of the two agents. The pursuer is generally located at [xp,yp] and the evader is generally located at [xe,ye]. Beginning at the initial positions, [xp0,yp0] for the pursuer and [xe0,ye0] for the evader, the pursuer should optimally drive the evader to a certain location or region. For the 1P1E scenario, this is the [0,0] position in the x-y plane.

To formulate this problem mathematically, we must create a model that governs the interaction of the agents. We require that the evader moves away from the pursuer according to two mathematical rules. First, the direction of the motion of the evader is along the straight line joining the pursuer and the evader. Second, the evader's velocity is bounded by the inverse of the distance between the evader and the pursuer. The dissertation by Shedied [9] assumes the distance between the pursuer and the evader is always the constant. This model is not very realistic either in the context of animal herding or robot motion. In contrast, the relative distance, r, in this thesis is allowed to be time varying within a set of limits. As the pursuer gets close to the evader, the evader increases its velocity in the direction opposite to the pursuer. This means the velocity is increasing while r is decreasing, but r has its own lower and upper bounds. For example we might specify that r must lie in the region [1,4]. The associated dynamics of the problem are given below:

$$\tan(\theta) = \frac{\dot{y}_e}{\dot{x}_e} \quad \text{(radial direction evasion policy)} \tag{2.1}$$

$$\text{where: } \theta = \tan^{-1}\left(\frac{y_p - y_e}{x_p - x_e}\right)$$

$$\dot{x}_e^2 + \dot{y}_e^2 = \frac{1}{r^2}, r \in [1,4] \quad (\tfrac{1}{r} \text{ evasion policy for speed}) \tag{2.2}$$

$$\text{where: } r \;=\; \sqrt{(x_p - x_e)^2 + (y_p - y_e)^2}$$

B. Formulation of Optimal Control Problem

The goal of the pursuer is to drive the evader from the given initial position to the final region. To formulate the herding problem as an optimal control problem, we define a performance index and determine the controls. We assume that previously documented evasion policy is known and deterministic, and choose the square of the velocity of the pursuer as a measure of the pursuer control effort because it is well suitated for the robot application. We may also consider the time duration of herding as another part of the performance index. The performance index J can be expressed in this form:

$$J = w_1 \cdot t_f + w_2 \cdot \int_0^{t_f} (\dot{x}_p^2 + \dot{y}_p^2) dt \tag{2.3}$$

In the above expression, w_1 and w_2 are the weights for time optimal and effort optimal respectively. As one can see, when $w_1 = 0$ and $w_2 = 1$, the final time is removed from the performance index. This corresponds to an effort optimal case. When $w_1 = 1$ and $w_2 = 0$, the control effort is not penalized, and any solution with respect to J becomes a time optimal solution. With this performance index J, the pursuer is to drive the evader from the given initial position to the desired destination following a changeable weighted time optimal and effort optimal path. Since $\dot{x}_p$ and $\dot{y}_p$ appear in the integral part of the above performance index, it is straightforward to use them as controls for the pursuer. Another option is to use the rate change of the relative distance $\dot{r}$ and position angle $\dot{\theta}$ as controls, but these include position and velocity information of both agents. Using $\dot{x}_p$ and $\dot{y}_p$ is easier to implement in an experimental setting. Additionally, we can keep r and θ as intermediate variables to influence the motion of the evader. In reality, the motion of the pursuer has its own limitation so we can include lower and upper limits on the control. The controls (speed of the pursuer) are restricted in the following manner:

$$\text{Controls}: \begin{cases} \dot{x}_p \in [-1, 1] \\ \dot{y}_p \in [-1, 1] \end{cases} \tag{2.4}$$

We determine differential equations for the states by manipulating and simplifying the above kinematic models. Equation (2.1) can be written as

$$\dot{y}_e = \dot{x}_e \tan(\theta) \tag{2.5}$$

By substituting $\dot{y}_e$ from equation (2.5) into equation (2.2) , we get

$$\dot{x}_e^2(1 + \tan^2(\theta)) = \frac{1}{r^2} \tag{2.6}$$

Utilizing $1 + \tan^2(\theta) = \frac{1}{\cos^2(\theta)}$, equation (2.6) becomes

$$\dot{x}_e^2 = \frac{\cos^2(\theta)}{r^2} \tag{2.7}$$

By performing similar operations on equation (2.2) in terms of $\dot{y}_p$, we find

$$\dot{y}_e^2 = \frac{\sin^2(\theta)}{r^2} \tag{2.8}$$

To get state equations, we need to take square root of the equations (2.7) and (2.8). Because the evader moves away from the pursuer, we only take the minus sign for $\dot{x}_e$ and $\dot{y}_e$. So state equations are:

$$\text{State Equations}: \begin{cases} \dot{x}_e &= -\frac{1}{r}\cos(\theta) \\ \dot{y}_e &= -\frac{1}{r}\sin(\theta) \end{cases} \tag{2.9}$$

$$\begin{aligned} \text{where: } r &= \sqrt{(x_p - x_e)^2 + (y_p - y_e)^2} \\ \theta &= \tan^{-1}\left(\frac{y_p - y_e}{x_p - x_e}\right) \end{aligned}$$

For a representative computation, the initial conditions ($t_0 = 0$) are randomly chosen: $[x_e(t_0), y_e(t_0)] = [3, 4]$, $[x_p(t_0), y_p(t_0)] = [4, 4.6]$. The final constraint at t_f is $[x_e(t_f), y_e(t_f)] = [0, 0]$.

C. Dynamic Programming (SNOPT)

Once the 1P1E herding problem is formulated as an optimal control problem, the next step is solving it. Because the problem is nonlinear and the final time t_f is not fixed, it cannot be solved in closed form. The problem will increase in complexity as we include more agents in this game (such as a system of two pursuers and multiple evaders). One way of solving the optimal control problem is to apply a numerical discretization procedure to the problem, which is known as the direct method. Direct methods for solving optimization problems have become extremely popular because of the ease and speed. After discretization, parameter optimization techniques are employed by way of a nonlinear programming (NLP) algorithm. The algorithm adopted here is a sequential quadratic programming (SQP) algorithm implemented in the SNOPT software [16] in the MATLAB environment. Here we give an overview of the steps required to utilize the SNOPT software. A general MATLAB code that calls the SNOPT solver can be organized into the three following programs:

1. The Main Program - This will call all the functions (subroutines) used to define the variables and constraint functions used by SNOPT, called the main solver, and perform any additional data manipulation or plotting that the user wishes to do. In addition, this is where any optional parameters will be set.

2. Variable Function - Before SNOPT can be called, a number of variables must be defined. These can be defined in one function, incorporated into the main function, or divided up into separate functions if needed. In these functions,

upper and lower bounds are placed on the variables and the constraint functions. For example, the initial conditions are equality constraints and the lower and upper limits on the controls are inequality constraints. As for the inequality constraint functions, we need to constrain the relative distance between the pursuer and evader, r, to lie in the region [1,4] by setting upper and lower bounds on r. In terms of the problem formulation, this becomes a constraint on our state variables.

3. User Function - This function defines the constraint equations, F, and the nonzero elements of the Jacobian matrix, G. The vector F also contains the objective function. While it does not matter where the objective function is placed in F, one must redefine the 'Objective row' option to the correct row. By default SNOPT assumes the objective function is contained in row 1 of F. The numerical discretization procedure is applied in this subroutine. There are a large number of discretization techniques available. For this thesis, the Trapezoidal method is used to integrate the differential equations because it is straightforward to implement. Here we present an overview of the trapezoidal method. Suppose the standard performance index is

$$J = \phi(\mathbf{x}(t_f), t_f) + \int_0^{t_f} [L(\mathbf{x}(t), \mathbf{u}(t), t)]dt \tag{2.10}$$

The states, $\mathbf{x}$(t), and controls, $\mathbf{u}$(t), are subject to the dynamic constraints:

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), t], t \in [t_0, t_f] \tag{2.11}$$

After numerical discretization, the computation variables for the trapezoidal method are the states and controls at the N+1 node points, i.e., $\{\mathbf{x}_1, \mathbf{u}_1, ..., \mathbf{x}_{N+1}, \mathbf{u}_{N+1}\}$. Because of the computational burden, we set the number of time steps as $N = 20$

unless there is a special requirement to increase the number of time steps. The dynamic equation constraints are formulated as:

$$\mathbf{x}_{k+1}-\mathbf{x}_k-\frac{h}{2}[\mathbf{f}(\mathbf{x}_k,\mathbf{u}_k,t_k)+\mathbf{f}(\mathbf{x}_{k+1},\mathbf{u}_{k+1},t_{k+1})]=\mathbf{0}, k=1,2,...,N \qquad (2.12)$$

where h is the time step size. The performance index is approximated as:

$$J\cong\phi(\mathbf{x}_{N+1},t_{N+1})+\sum_{k=1}^{N}\frac{h}{2}\left\{L(\mathbf{x}_k,\mathbf{u}_k,t_k)+L(\mathbf{x}_{k+1},\mathbf{u}_{k+1},t_{k+1})\right\} \qquad (2.13)$$

So the trapezoidal method is used to approximately integrate the area below the $L(x)$ function. Figure 2 shows the picture of this method when the node points are from $N=2$ to $N=4$.

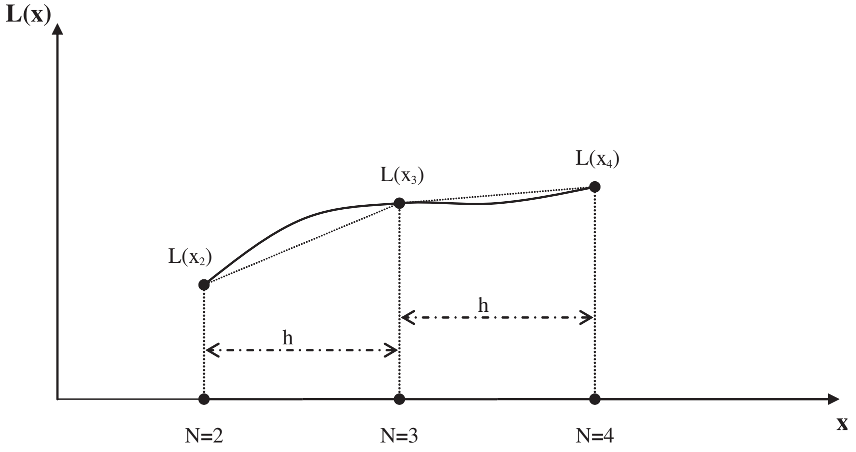

Fig. 2. Trapezoidal Method

D. Results of Designed Trajectories

When the weights $w_1=1$ and $w_2=1$, it becomes a time and effort optimal case. The trajectories of the pursuer and the evader can be seen in Figure 3. It demonstrates that the pursuer drives the evader to the goal ([0,0] position), in $t_f = 6.1963$ seconds. The value of the performance index J is 9.7651. The computation time is less than 2 seconds. Figure 4 shows the corresponding controls and relative distance and angle time histories. The responses all demonstrate smooth behavior. As one can see in Figure 4, the controls and relative distance are bounded by [-1,1] and [1,4] respectively.

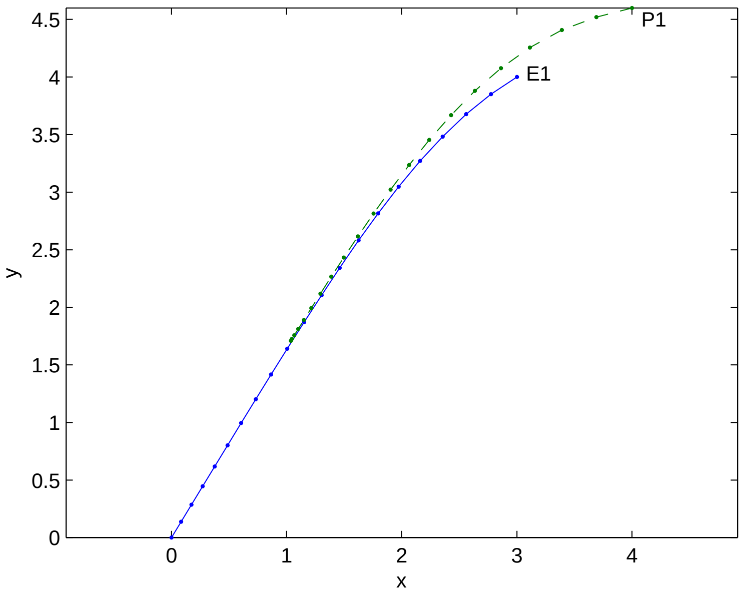

Fig. 3. Designed Trajectories for 1P1E and Time and Effort Optimal

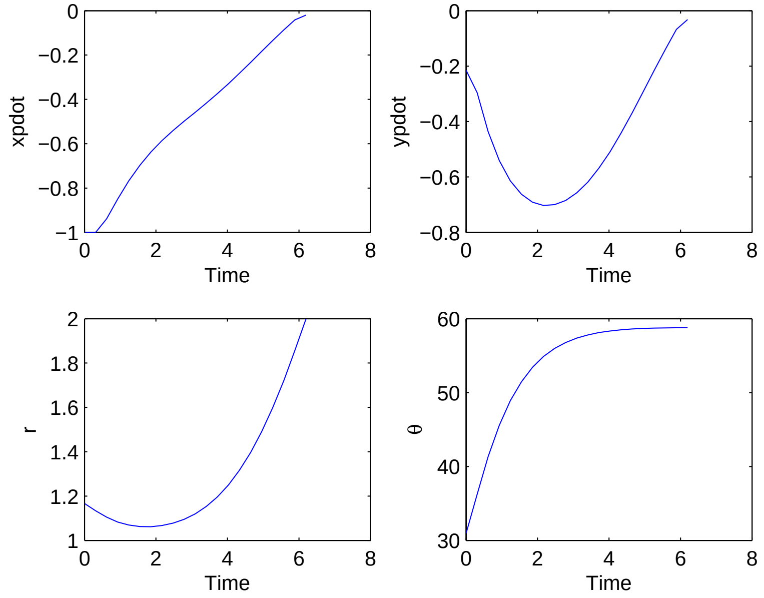

Fig. 4. Control and Relative Vector Time Histories for 1P1E and Time and Effort Optimal

E. Simulation Process and Results

Once all the trajectories of the pursuer and the evader are determined, then the motion of the entire system has to be simulated as an experiment. The pursuer moves according to the trajectory defined by offline computation, and the evader is completely reactionary. The evader acts as a cheap robot, reacting to the relative distance, r, and relative angle, θ, with respect to the pursuer. So the basic concept of the simulation is to solve the differential state equations of the evader by utilizing Ode45 (a Differential Equation Solver in the MATLAB) with the given initial condition of evader and the ideal trajectory of pursuer, $x_p(t)$ and $y_p(t)$. This simulation process is listed as the following:

- Initial Condition: $x_e(t_0), y_e(t_0)$

- Given: $x_p(t), y_p(t)$
- Solve: State Equations (2.9)

The given trajectory $x_p(t)$ and $y_p(t)$ as computed by solving the NLP problem is comprised of $N + 1$ points and is not continuous. Since we are solving the same state equations, ideally the simulated trajectory of evader should be identical to the designed offline trajectory. However, Ode45 uses a high-order numerical integration method while the trapezoidal method is a low-order integration method. This may result in differences between the simulated and the designed offline trajectories. Figure 5 demonstrates the simulated trajectory of the evader along with the designed offline trajectories for the case where $w_1 = 1$ and $w_2 = 1$, the time and effort optimal case. One can see that the simulated trajectory closely resembles the designed offline trajectory. For the following cases and scenarios, the simulated and the designed offline trajectories are plotted in the same figure. For the remainder of thesis, when both trajectories match very well, a legend will not be shown in the plot.

When $w_1 = 0$ and $w_2 = 1$ (the effort optimal case) $t_f = 14.1352$ and $J = 0.5309$. Figure 6 shows that the simulated trajectory of the evader slightly diverges the designed trajectory. The pursuer trajectory is relatively curved because the pursuer is driving the evader with the minimum amount of control effort, time is not optimized because the first weight is set as $w_1 = 0$. Also, Figure 7 shows that the magnitudes of the controls are relatively small compared to other cases.

The time optimal case occurs when $w_1 = 1$ and $w_2 = 0$. The corresponding values of the final time and performance index are $t_f = 5.1189$ and $J = 5.1189$ respectively. Figure 8 and Figure 9 demonstrate the results for the time optimal case. The trajectory seems to follow a straight line after a maneuver because the pursuer must drive the evader to the goal as soon as possible. The maneuver requires a high

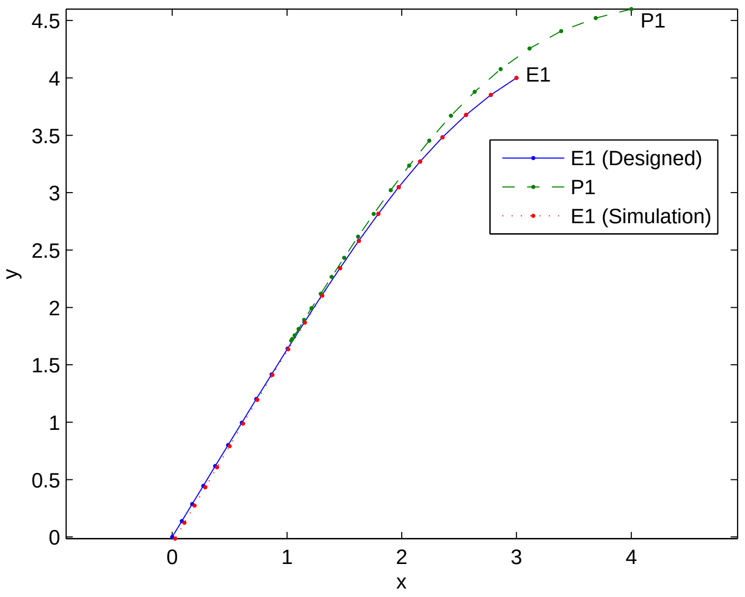

Fig. 5. Designed and Simulated Trajectories for 1P1E and Time and Effort Optimal

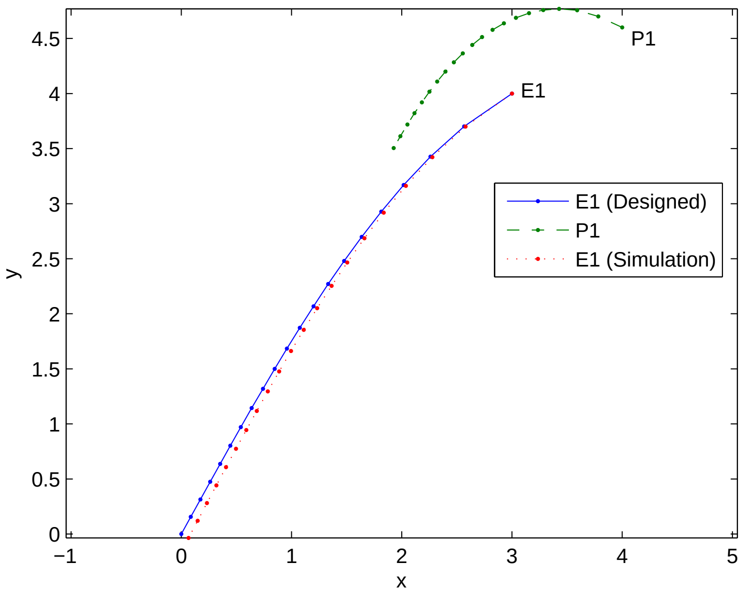

Fig. 6. Designed and Simulated Trajectories for 1P1E and Effort Optimal

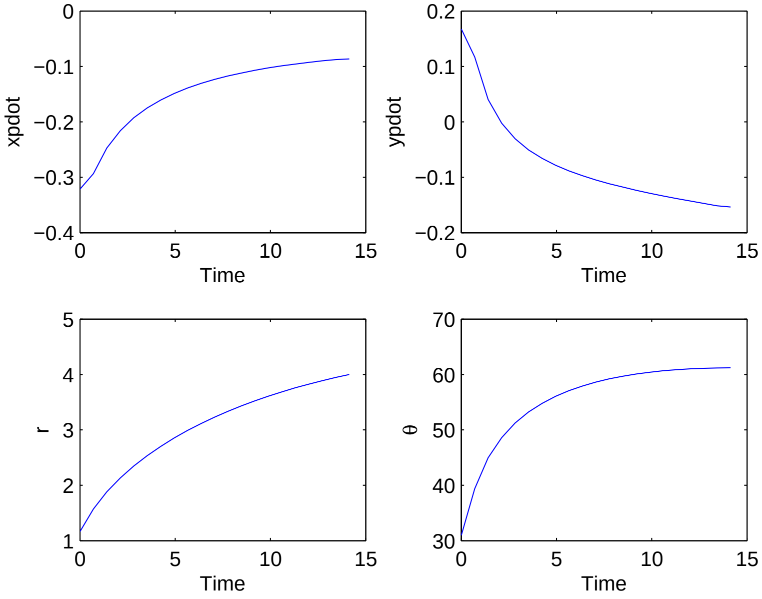

Fig. 7. Control and Relative Vector Time Histories for 1P1E and Effort Optimal

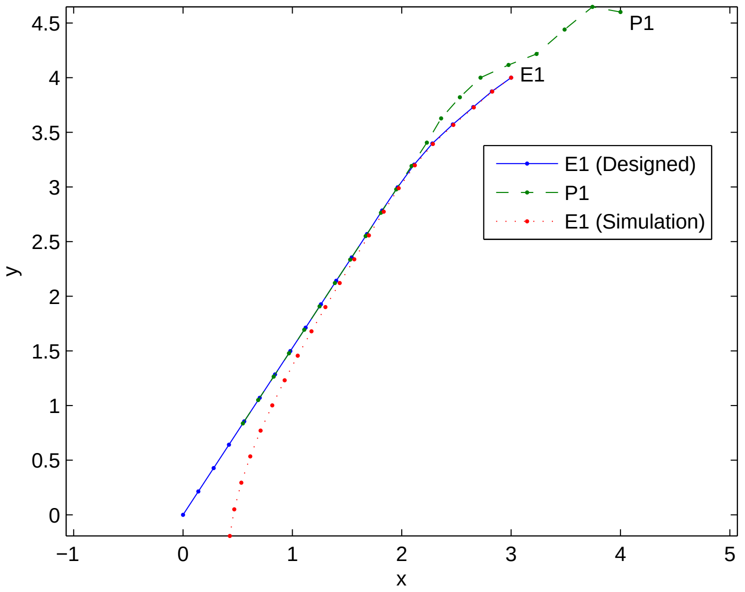

Fig. 8. Designed and Simulated Trajectories for 1P1E and Time Optimal

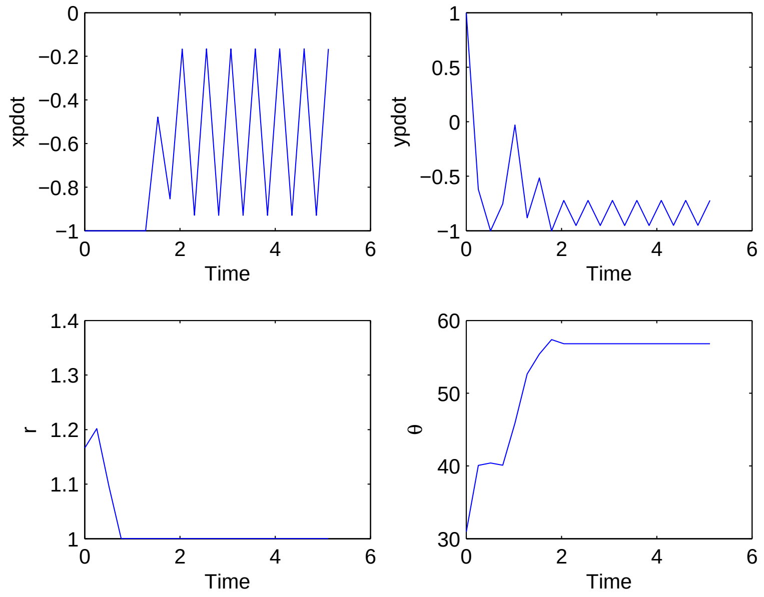

Fig. 9. Control and Relative Vector Time Histories for 1P1E and Time Optimal

amount of control to change the evader's orientation so that it will follow the straight line behavior while the pursuer is pushing the evader. The pursuer approaches the evader until the relative distance, r, reaches its lower bound. One can see that r and θ in Figure 9 becomes constant after around 1.9 seconds when agents are moving in a straight line after an orientation maneuver. If we examine equations (2.9) and (2.2), we see that for constant r and θ, the controls should be constant as well. However, Figure 9 shows that the controls are chattering. This is caused by numerical error. Figure 8 shows that the simulated trajectory of the evader diverges from the designed trajectory. In order to make this discrepancy small, we can increase the number of discrete node points from $N = 20$ to $N = 50$. Figure 10 shows that the simulated and designed trajectories of the evader match fairly well until the later part (last 9 points). The disadvantage of increasing the number of points is an increase in computational burden. For example, 50 points takes near 6 seconds of computation time.

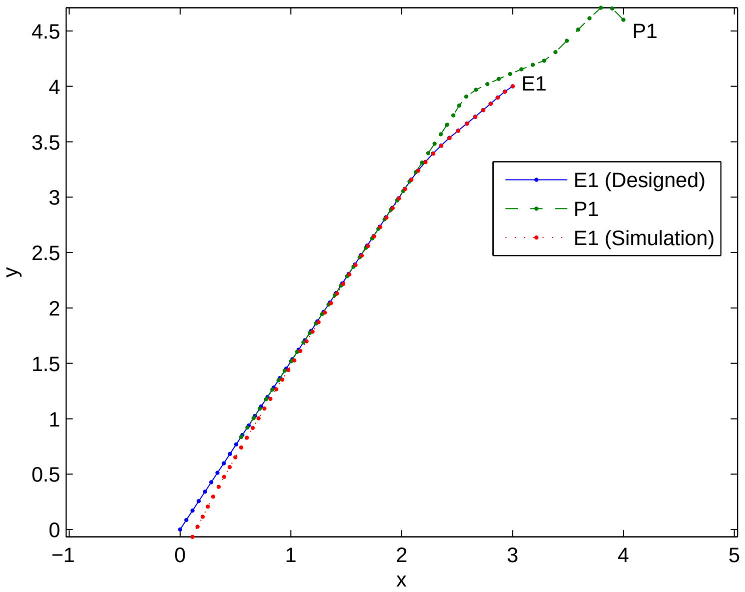

Fig. 10. Designed and Simulated Trajectories for 1P1E and Time Optimal (50 Points)

F. Conclusion and Discussion

In this chapter, we introduced the one pursuer and one evader simple system (1P1E scenario). A mathematical model was developed that governs the motion of the evader away from the pursuer. Using this model, we formulated a basic herding problem as an optimal control problem by defining state equations, controls, constraints and a performance index. Given random initial conditions, the designed offline trajectories of two agents are solved numerically with N discrete time steps by utilizing the SNOPT software in the MATLAB environment to solve the resulting nonlinear programming problem. By solving the differential state equations, we then simulate the motion of the system with the pursuer following the trajectory determined by solving the optimal control problem, and with the evader reacting to the relative distance and relative angle to the pursuer. The results of three cases, created by changing the weights in the performance index (time and effort optimal, effort optimal and time optimal) are considered. As expected, the time duration of the herding, t_f, is the largest for the effort optimal case and the smallest for the time optimal case. The performance index J is the highest in the time and effort optimal case because it is involving both terms. In the effort optimal case, J is the lowest, because the pursuer is driving the evader with the minimum amount of control effort regardless of time as long as it satisfies the final constraint. Inevitably, there are numerical errors in the time optimal case as seen in Figure 9. This was discussed in the last section. When simulating the response of the system, a high order numerical integration method was used such as MATLAB's Ode45. Despite the difference in integration method, the designed offline and simulated trajectories match well in all cases except the time optimal case. This case can be improved by increasing the number of discrete points from $N = 20$ to $N = 50$. Additionally, the time optimal case requires more computation time (2.5006 seconds) than other cases (less than 2 seconds). The time optimal control case is computationally more challenging than the other cases.

CHAPTER III

TWO PURSUERS AND TWO OR MULTIPLE EVADERS SYSTEM

In the previous chapter a game involving two agents, one pursuer and one evader, was discussed. We now address the more challenging problem of a herding game involving groups of pursuers and evaders. Methodologies for addressing this problem will be considered in this chapter.

A. Two Sets of Independent One Pursuer and One Evader Plus Another Evader Scenario (2×1P1E+E3)

For this system, we introduce five agents: Pursuer 1 (P1) and Pursuer 2 (P2), which represent expensive robots with global navigation and relative position sensors, and Evader (E1), Evader 2 (E2) and Evader 3 (E3), which represent a team of three relatively cheap robots with only relative position sensors.

1. System Model and Formulation of Optimal Control Problem

Using the same mathematical model as the 1P1E scenario, we develop two sets of independent 1P1E models first. This means there is no cooperation between P1 and P2 in the game. As shown in Figure 11, each pursuer only deals with the evader with the same number. There is no influence of P1 on E2 or P2 on E1. Player E3 will be considered in the simulation part later. However, since we are seeking to have E1 and E2 arrive at the goal at the same time, the performance index should thus include both independent problems. Hence, with this system, the performance index is

$$J = w_1 \cdot t_f + w_2 \cdot \int_0^{t_f} (\dot{x}_{p1}^2 + \dot{y}_{p1}^2 + \dot{x}_{p2}^2 + \dot{y}_{p2}^2)dt \tag{3.1}$$

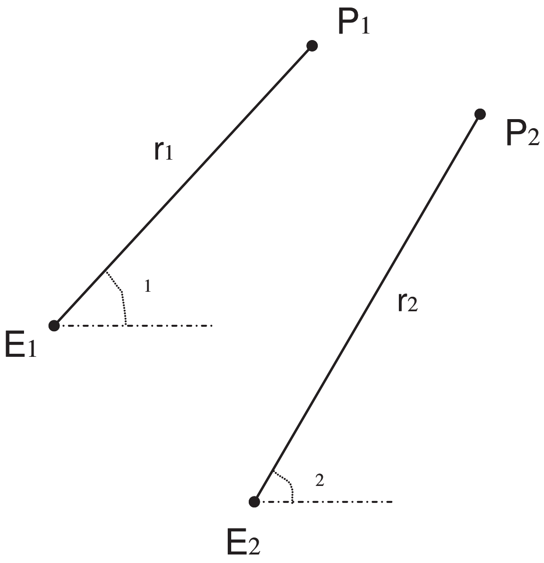

Fig. 11. System Model for 2×1P1E

The time optimal part is the same as the 1P1E problem and the effort optimal part is the summation of both pursuers. The state equations and controls in this scenario will be represented by two sets of independent 1P1E models as the following:

$$\mathrm{E}_1 : \begin{cases} \dot{x}_{e1} &= -\frac{1}{r_1}\cos(\theta_1) \\ \dot{y}_{e1} &= -\frac{1}{r_1}\sin(\theta_1) \end{cases} \tag{3.2}$$

$$\begin{aligned} \text{where: } r_1 &= \sqrt{(x_{p1}-x_{e1})^2+(y_{p1}-y_{e1})^2} \\ \theta_1 &= \tan^{-1}\left(\frac{y_{p1}-y_{e1}}{x_{p1}-x_{e1}}\right) \end{aligned}$$

$$\mathrm{E}_2 : \begin{cases} \dot{x}_{e2} &= -\frac{1}{r_2}\cos(\theta_2) \\ \dot{y}_{e2} &= -\frac{1}{r_2}\sin(\theta_2) \end{cases} \tag{3.3}$$

$$\text{where: } r_2 = \sqrt{(x_{p2}-x_{e2})^2+(y_{p2}-y_{e2})^2}$$

$$\theta_2 = \tan^{-1}\left(\frac{y_{p2} - y_{e2}}{x_{p2} - x_{e2}}\right)$$

$$\text{Controls :} \begin{cases} \dot{x}_{p1} \in [-1, 1] \\ \dot{y}_{p1} \in [-1, 1] \\ \dot{x}_{p2} \in [-1, 1] \\ \dot{y}_{p2} \in [-1, 1] \end{cases} \tag{3.4}$$

For the examples we will consider, the initial conditions ($t_0 = 0$) are: $[x_{e1}(t_0), y_{e1}(t_0)] = [2.8, 4.1]$, $[x_{e2}(t_0), y_{e2}(t_0)] = [3.5, 3.5]$, $[x_{p1}(t_0), y_{p1}(t_0)] = [5, 5.6]$, and $[x_{p2}(t_0), y_{p2}(t_0)] = [5.4, 5.1]$ (chosen randomly).

The final constraint at t_f is $[x_{e1}(t_f), y_{e1}(t_f)] = [0, 0]$ and $[x_{e2}(t_f), y_{e2}(t_f)] = [0, 0]$.

2. Results of Designed Trajectories

When $w1 = 1$ and $w_2 = 1$ (the time and effort optimal case), Figure 12 shows that each pursuer drives its own evader (with the same number) to the goal ([0,0] position) in time $t_f = 10.7456$ seconds. The value of the performance index, J, is 15.8639. The computation time is about 5.5 seconds. Figure 13 shows the corresponding controls, relative distance, and relative angle time histories. The responses are very similar to the 1P1E scenario.

3. Simulation Process and Results with an Additional E3

The simulation of this scenario is similar to the 1P1E scenario, with the major difference being that we have two sets of 1P1E. Furthermore, we can include an additional Evader, E3, in the scenario as shown in Figure 14.

The extra evader, E3, reacts to the motion of either P1 or P2. The simulation process is listed as following:

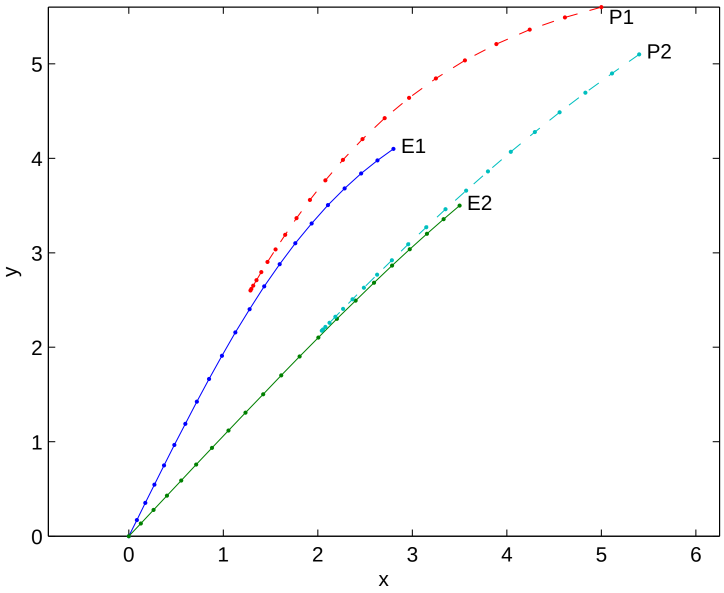

Fig. 12. Designed Trajectories for 2×1P1E and Time and Effort Optimal

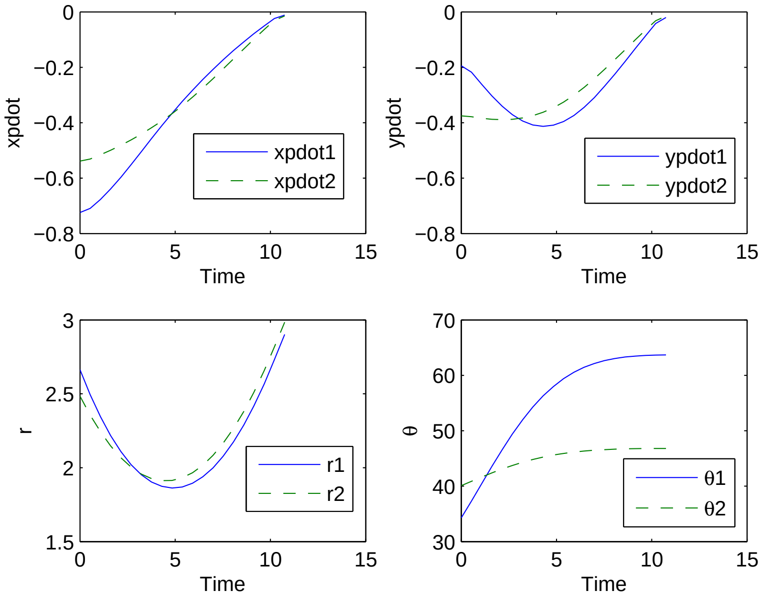

Fig. 13. Control and Relative Vector Time Histories for 2×1P1E and Time and Effort Optimal

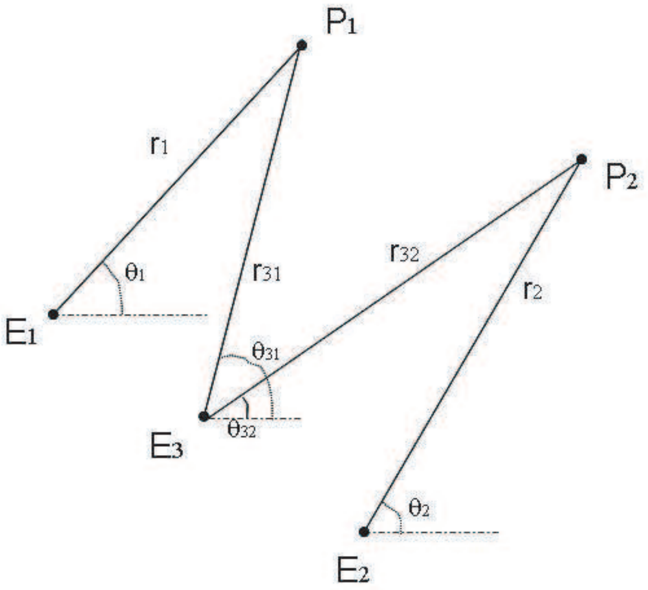

Fig. 14. System Model for 2×1P1E+E3

- Initial Condition: $[x_{e1}(t_0), y_{e1}(t_0)]$, $[x_{e2}(t_0), y_{e2}(t_0)]$ and $[x_{e3}(t_0), y_{e3}(t_0)]$
- Given: $x_{p1}(t), y_{p1}(t), x_{p2}(t), y_{p2}(t)$
- Solve: State Equations (3.2), (3.3) and the following equation (3.5) for the E_3 part.

$$\mathrm{E}_3 : \begin{cases} \dot{x}_{e3} & = -\frac{1}{r_3}\cos(\theta_3) \\ \dot{y}_{e3} & = -\frac{1}{r_3}\sin(\theta_3) \end{cases} \tag{3.5}$$

where: (r_3 and θ_3 chosen based on which pursuer (P1 or P2) drives E3)

$$\begin{aligned} r_3 &= \sqrt{(x_{p1} - x_{e3})^2 + (y_{p1} - y_{e3})^2} \\ \theta_3 &= \tan^{-1}\left(\frac{y_{p1} - y_{e3}}{x_{p1} - x_{e3}}\right) \\ \text{or: } r_3 &= \sqrt{(x_{p2} - x_{e3})^2 + (y_{p2} - y_{e3})^2} \\ \theta_3 &= \tan^{-1}\left(\frac{y_{p2} - y_{e3}}{x_{p2} - x_{e3}}\right) \end{aligned}$$

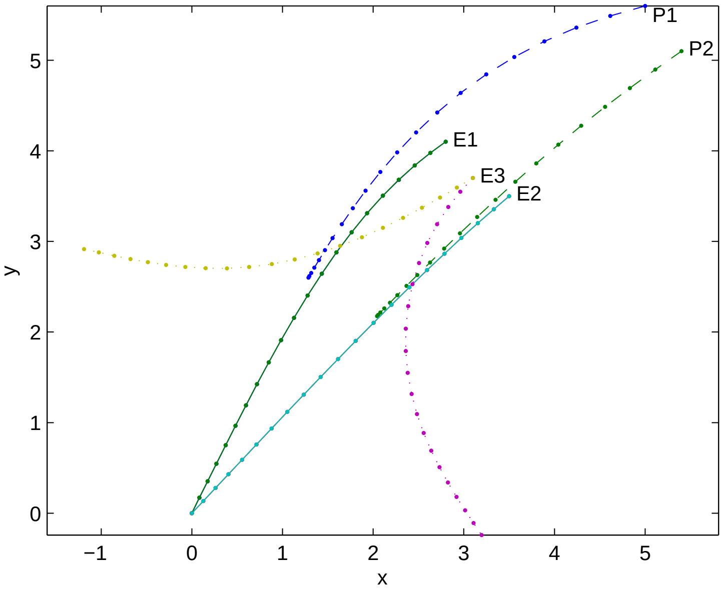

Fig. 15. Designed and Simulated Trajectories for 2×1P1E+E3 and Time and Effort Optimal

Figure 15 shows that the simulated trajectories of E1 and E2 match the designed offline trajectories. E3 (in this case $[x_{e3}(t_0), y_{e3}(t_0)] = [3.1, 3.7]$) is not approaching the goal. Instead, it goes from away from either P1 or P2 depending on which pursuer E3 reacts to.

4. Conclusion and Discussion

In this section, we combined two sets of independent 1P1E together in the same performance index using the 1P1E scenario that was proposed in the last chapter. The whole process, which involves using random initial conditions to solve the designed offline trajectories and then simulating the motion of the system, is similar to the 1P1E scenario. Each pursuer is able to drive its own evader to the goal at the same time. The point of this scenario is to demonstrate how the evaders react when pursuers do not cooperate. As shown in Figure 15, the trajectory of each single evader is only influenced by the trajectory of its corresponding pursuer. Without cooperation between P1 and P2, one can see there is no way for P1 and P2 to handle E3 in addition to their own evader. Neither of the pursuers is able to independently drive E3 to the goal. Incorporating this into our scenario increases the complexity of the problem. This raises the question, how can two pursuers cooperate to drive multiple evaders to a certain location or region in a manner that is weighted time optimal and effort optimal?

B. Two Pursuers and Three Evaders Scenario (2P3E)

In this section, we will develop a mathematical cooperation model that allows two pursuers to drive a team of three evaders to the goal at the same time.

1. System Model

A diagram of the scenario is shown in Figure 16. There are six relative distance vectors relating the two pursuers to the three evaders as well as six relative angles. The motion of each evader is influenced by the relative vector of each pursuer. This can be thought of as an “influence” from each pursuer. If we want both pursuers to influence the motion of a single evader, we can consider the summation of both relative vectors, or influence, from each pursuer. This idea is illustrated in Figure 17. The mathematical model is given below:

$$\underline{F_i} = \underline{F_{i1}} + \underline{F_{i2}} \tag{3.6}$$

$$\text{In x direction} : F_{ix} = F_{i1x} + F_{i2x} \tag{3.7}$$

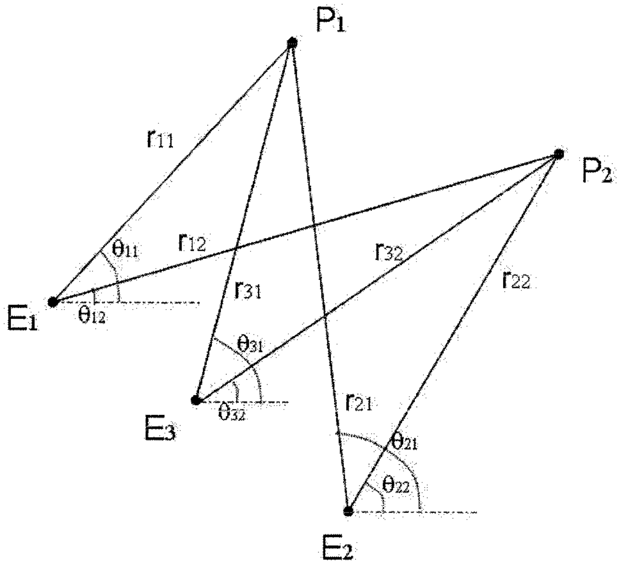

Fig. 16. System Model for 2P3E

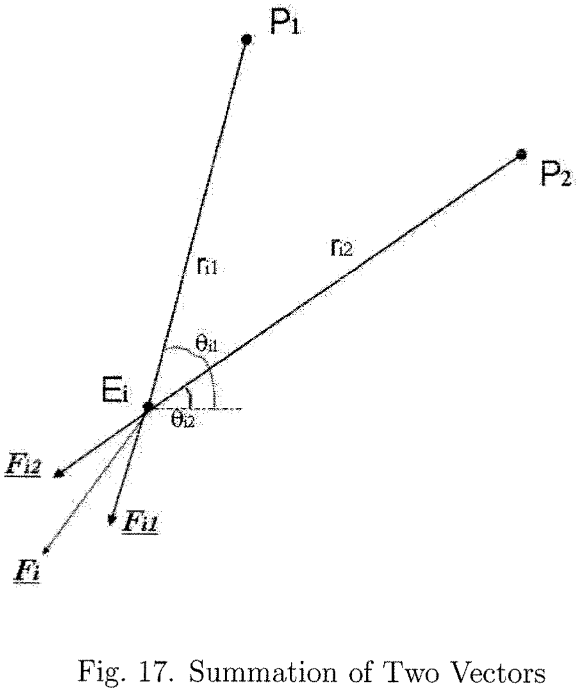

Fig. 17. Summation of Two Vectors

$$\text{In y direction} : F_{iy} \quad = \quad F_{i1y} + F_{i2y} \tag{3.8}$$

$$\begin{aligned} \text{where: } F_{i1x} &= -\frac{1}{r_{11}}\cos(\theta_{11}) \\ F_{i2x} &= -\frac{1}{r_{12}}\cos(\theta_{12}) \\ F_{i1y} &= -\frac{1}{r_{11}}\sin(\theta_{11}) \\ F_{i2y} &= -\frac{1}{r_{12}}\sin(\theta_{12}) \end{aligned}$$

2. Formulation of Optimal Control Problem

With the influence model presented in equations (3.6) to (3.8), we can develop the dynamic equations that enable cooperation between pursuers. The state equations are given by:

$$\mathrm{E}_1 : \begin{cases} \dot{x}_{e1} &= -\frac{1}{r_{11}}\cos(\theta_{11}) - \frac{1}{r_{12}}\cos(\theta_{12}) \\ \dot{y}_{e1} &= -\frac{1}{r_{11}}\sin(\theta_{11}) - \frac{1}{r_{12}}\sin(\theta_{12}) \end{cases} \tag{3.9}$$

$$\begin{aligned} \text{where: } r_{11} &= \sqrt{(x_{p1} - x_{e1})^2 + (y_{p1} - y_{e1})^2} \\ \theta_{11} &= \tan^{-1}\left(\frac{y_{p1} - y_{e1}}{x_{p1} - x_{e1}}\right) \\ r_{12} &= \sqrt{(x_{p2} - x_{e1})^2 + (y_{p2} - y_{e1})^2} \\ \theta_{12} &= \tan^{-1}\left(\frac{y_{p2} - y_{e1}}{x_{p2} - x_{e1}}\right) \end{aligned}$$

$$\mathrm{E}_2 : \begin{cases} \dot{x}_{e2} &= -\frac{1}{r_{21}}\cos(\theta_{21}) - \frac{1}{r_{22}}\cos(\theta_{22}) \\ \dot{y}_{e2} &= -\frac{1}{r_{21}}\sin(\theta_{21}) - \frac{1}{r_{22}}\sin(\theta_{22}) \end{cases} \tag{3.10}$$

$$\text{where: } r_{21} \quad = \quad \sqrt{(x_{p1} - x_{e2})^2 + (y_{p1} - y_{e2})^2}$$

$$\begin{aligned}\theta_{21} &= \tan^{-1}\left(\frac{y_{p1}-y_{e2}}{x_{p1}-x_{e2}}\right)\\ r_{22} &= \sqrt{(x_{p2}-x_{e2})^2+(y_{p2}-y_{e2})^2}\\ \theta_{22} &= \tan^{-1}\left(\frac{y_{p2}-y_{e2}}{x_{p2}-x_{e2}}\right)\end{aligned}$$

$$\mathrm{E}_3 : \begin{cases} \dot{x}_{e3} &= -\frac{1}{r_{31}}\cos(\theta_{31}) - \frac{1}{r_{32}}\cos(\theta_{32}) \\ \dot{y}_{e3} &= -\frac{1}{r_{31}}\sin(\theta_{31}) - \frac{1}{r_{32}}\sin(\theta_{32}) \end{cases} \tag{3.11}$$

$$\begin{aligned}\text{where: } r_{31} &= \sqrt{(x_{p1}-x_{e3})^2+(y_{p1}-y_{e3})^2}\\ \theta_{31} &= \tan^{-1}\left(\frac{y_{p1}-y_{e3}}{x_{p1}-x_{e3}}\right)\\ r_{32} &= \sqrt{(x_{p2}-x_{e3})^2+(y_{p2}-y_{e3})^2}\\ \theta_{32} &= \tan^{-1}\left(\frac{y_{p2}-y_{e3}}{x_{p2}-x_{e3}}\right)\end{aligned}$$

In the state equations (3.9) to (3.11), the first subscript of r and θ corresponds to the evader and the second one corresponds to the pursuer. As in previous scenarios, we keep $\dot{x}_p$ and $\dot{y}_p$ as controls as seen in equations (3.12). We use $\dot{x}_p$ and $\dot{y}_p$ as controls rather than $\dot{r}$ and $\dot{\theta}$ because using the relative distances and angles would require the addition of eight control constraints to the problem because the r_{ij} and θ_{ij} are not all independent. Each control obeys the constraints used in previous examples. Those are given by:

$$\text{Controls} : \begin{cases} \dot{x}_{p1} \in [-1, 1] \\ \dot{y}_{p1} \in [-1, 1] \\ \dot{x}_{p2} \in [-1, 1] \\ \dot{y}_{p2} \in [-1, 1] \end{cases} \tag{3.12}$$

For this scenario, we use the same performance index as the last 2×1P1E scenario. The constraint on relative distance coupled with other constraints gives 16×N+7 constraint functions when converted to the nonlinear programming problem. To help reduce the complexity, we incorporate soft constraints into the herding problem. Instead of keeping the final constraints, $x_e(t_f) = 0$ and $y_e(t_f) = 0$, we incorporate soft final constraints, $x_e^2(t_f) + y_e^2(t_f)$, into the performance index with a weight w_3. In this manner, the final goal of the team of evaders becomes a small region, like a pen in the context of animal herding, but it is very close to the origin. The performance index J can be expressed in the following manner:

$$J = w_1 \cdot t_f + w_2 \cdot \int_0^{t_f} (\dot{x}_{p1}^2 + \dot{y}_{p1}^2 + \dot{x}_{p2}^2 + \dot{y}_{p2}^2) dt$$
$$+ w_3 \cdot (x_{e1}^2(t_f) + y_{e1}^2(t_f) + x_{e2}^2(t_f) + y_{e2}^2(t_f) + x_{e3}^2(t_f) + y_{e3}^2(t_f)) \tag{3.13}$$

In this performance index we must choose a proper value of weight (for our example choose $w_3 = 10$ unless there is a special requirement to change the weight) to penalize the third term.

For our example, we choose the initial conditions to be: $[x_{e1}(t_0), y_{e1}(t_0)] = [2.8, 4.1]$, $[x_{e2}(t_0), y_{e2}(t_0)] = [3.5, 3.5]$, $[x_{e3}(t_0), y_{e3}(t_0)] = [3.1, 3.7]$, $[x_{p1}(t_0), y_{p1}(t_0)] = [5, 5.6]$ and $[x_{p2}(t_0), y_{p2}(t_0)] = [5.4, 5.1]$(chosen randomly).

3. Results of Designed Trajectories

Now that we have formulated our optimal control problem, we can determine the optimal trajectories. Figure 18 demonstrates that two pursuers successfully drive a team of three evaders to the goal for the time and effort optimal case ($w_1 = 1$ and $w_2 = 1$) in $t_f = 9.3388$ seconds. The value of the performance index J is 22.0369 and the computation time is near 16 seconds. Figure 19 shows the corresponding controls, relative distance, and relative angle time histories. The responses all demonstrate smooth behavior.

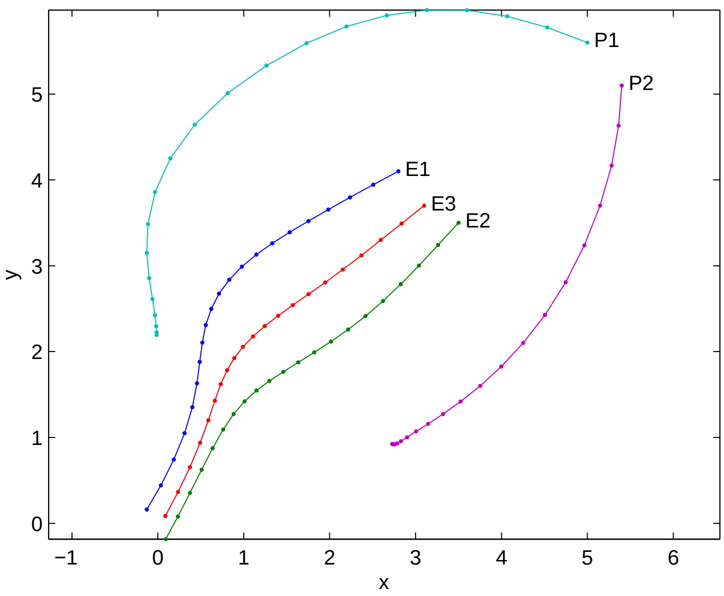

Fig. 18. Designed Trajectories for 2P3E and Time and Effort Optimal

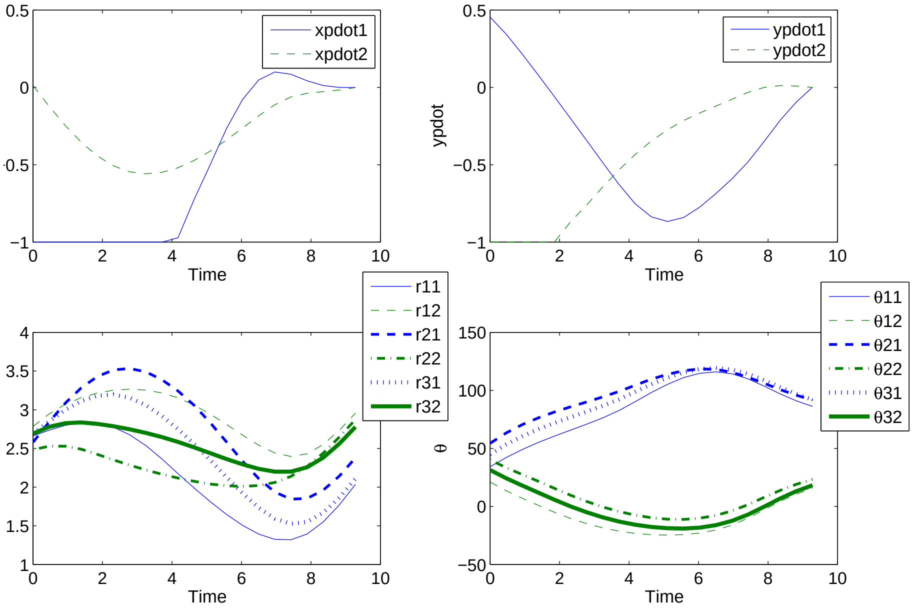

Fig. 19. Control and Relative Vector Time Histories for 2P3E and Time and Effort Optimal

4. Simulation Process and Results

With this cooperative model, this simulation process is explained by the following:

- Initial Condition: $[x_{e1}(t_0), y_{e1}(t_0)]$, $[x_{e2}(t_0), y_{e2}(t_0)]$ and $[x_{e3}(t_0), y_{e3}(t_0)]$ (Same as the 2×1P1E+E3 scenario)
- Given: $x_{p1}(t), y_{p1}(t), x_{p2}(t), y_{p2}(t)$ (Same as the 2×1P1E+E3 scenario)
- Solve: State Equations from (3.9) to (3.11)

All three cases, including time and effort optimal, effort optimal and time optimal, are computed and simulated in this scenario. For convenience, we list the results of three cases in Table I.

Table I. Results for 2P3E scenario

	Time and Effort Optimal	Effort Optimal	Time Optimal
t_f	9.3388	19.095	7.1002
J	22.0369	9.9193	7.4053
Computation Time	16.3764	64.8371	34.9583

For the three cases, the simulated and designed offline trajectories are plotted in the same figure and are very consistent to each other, as shown in Figures 20, 21 and 22. J is the highest when $w_1 = 1$ and $w_2 = 1$ (time and effort optimal case). When $w_1 = 0$ and $w_2 = 1$ (effort optimal case), Figure 23 shows that the magnitudes of the controls are relatively small and are approaching 0 near 10 seconds. For this case, the relative distances are relatively high (near the upper bound) because the pursers do not use as much effort. As a result, the evaders move away from the pursuers until

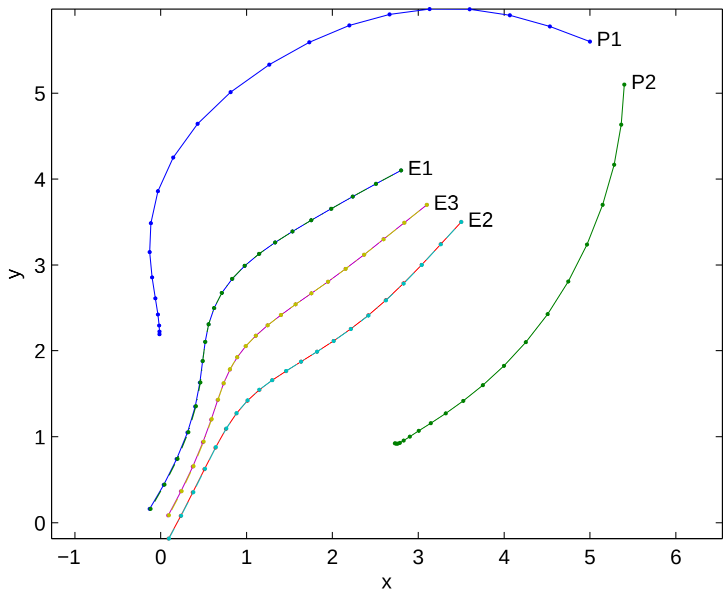

Fig. 20. Designed and Simulated Trajectories for 2P3E and Time and Effort Optimal

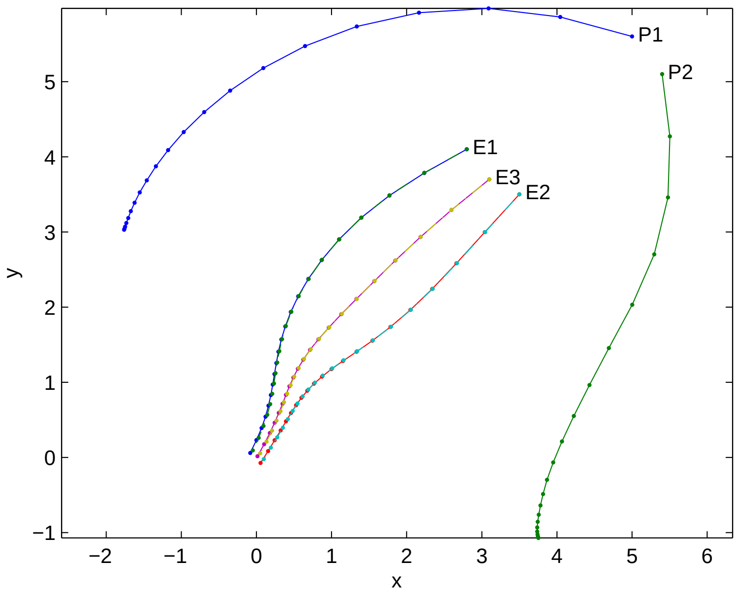

Fig. 21. Designed and Simulated Trajectories for 2P3E and Effort Optimal

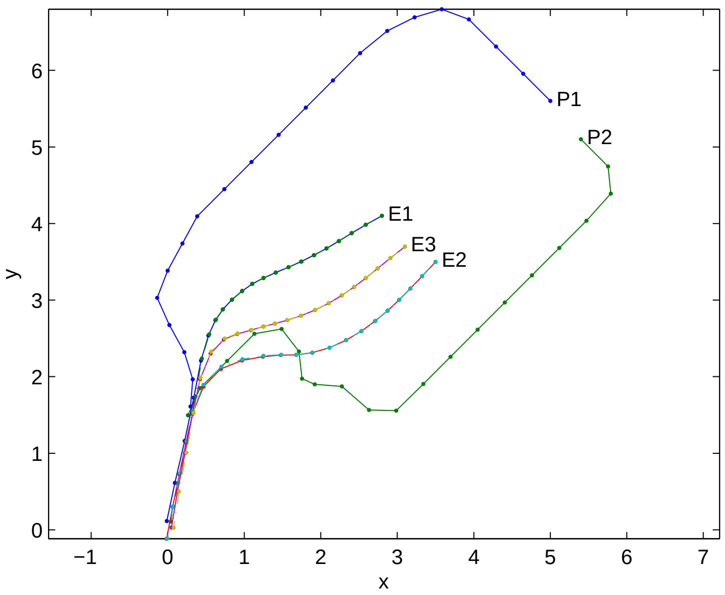

Fig. 22. Designed and Simulated Trajectories for 2P3E and Time Optimal

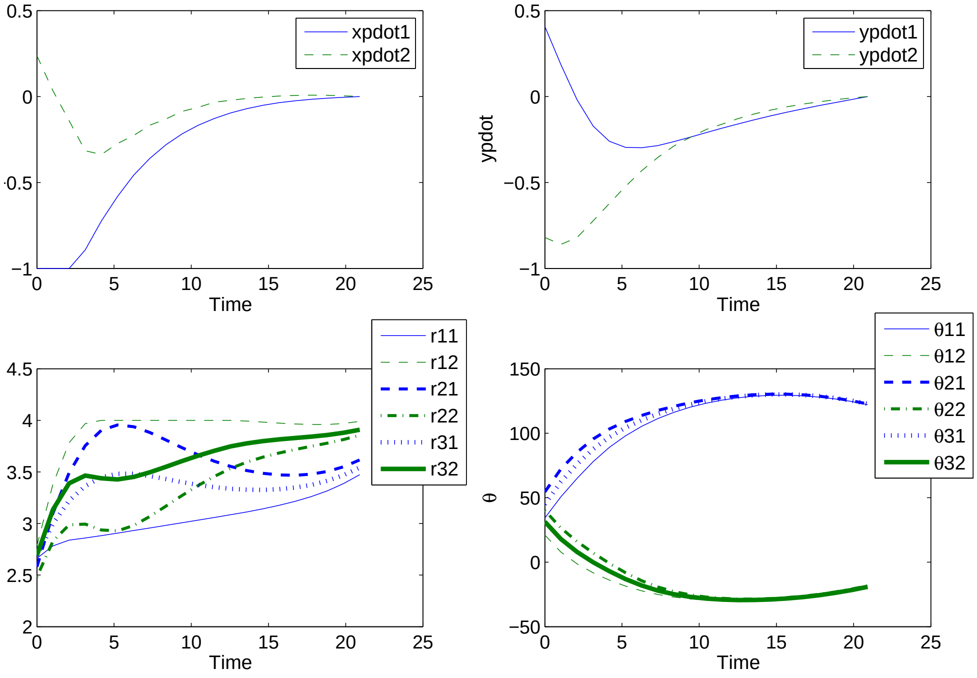

Fig. 23. Control and Relative Vector Time Histories for 2P3E and Effort Optimal

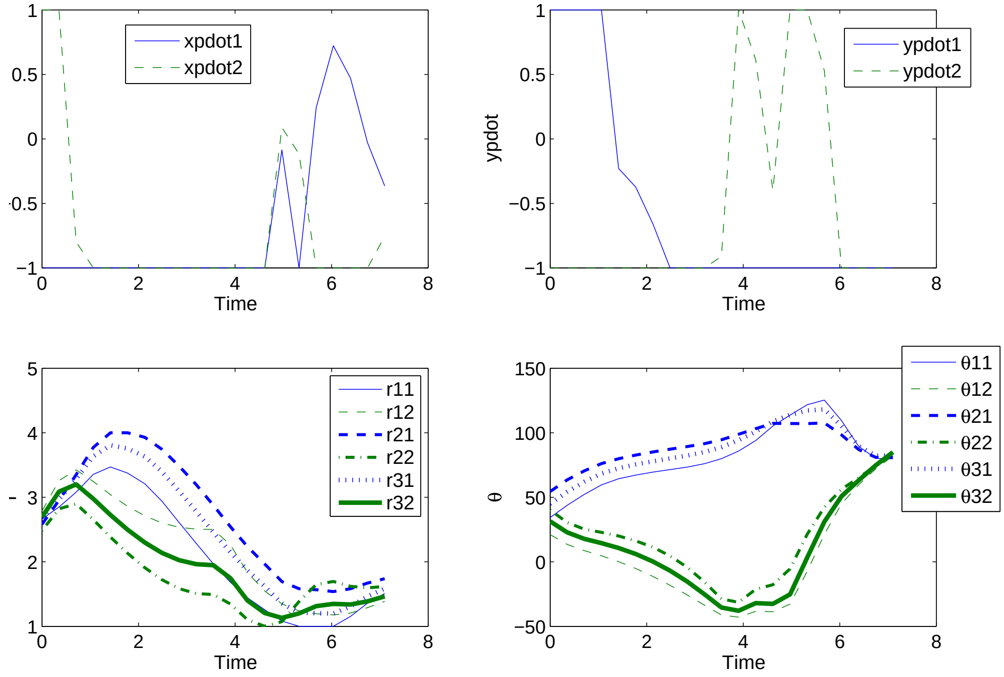

Fig. 24. Control and Relative Vector Time Histories for 2P3E and Time Optimal

the upper limit is reached. This causes the herding for this case to take the longest time.When $w_1 = 1$ and $w_2 = 0$ (time optimal case), the herding takes the shortest time as expected, Figure 24 shows that the controls for this case are not smooth, but display a chattering response.

5. Conclusion and Discussion

In this section, a new mathematical cooperation model was developed that governs the interaction between pursuers and evaders. We formulated this model as an optimal control problem using the same process as in the previous scenarios, with the exception that the final goal became a small region instead of the origin. This was accomplished by penalizing the final value of $x_e(t_f)$ and $y_e(t_f)$ (using soft constraints) instead of using the hard constraints, $x_e(t_f) = 0$ and $y_e(t_f) = 0$. After solving the designed offline trajectories, the results show that two pursuers are able to successfully drive a team of three evaders to the goal for three cases, created by changing the weights in the performance index. That displays the benefit of the cooperation in this scenario. The results are straightforward and meet our expectations, with the exception of the time optimal case. The control response for this case chatters, which is typical of time optimal problems. The computation time is one magnitude higher than the 1P1E scenario because there are 16×N+7 constraint functions in the problem even without the final constraints.

C. Two Pursuers and Two Evaders Scenario (2P2E)

Even though using two pursuers to drive an additional E3 to the goal is the primary objective of cooperation, we would also like to see how the performance index is improved with the cooperative model compared to the scenario using two sets of independent 1P1E (2×1P1E). Since the 2×1P1E scenario deals only with moving two evaders toward the goal at the same time, we consider two pursuers cooperatively driving only two of the evaders in this section in order to make the two scenarios comparable. Additionally, we would like to observe the response of an additional evader, E3, when it is included in the simulation and compare this to the 2P3E scenario.

1. System Model, Formulation of Optimal Control Problem and Simulation Process

Obviously, for this case, the state equations, performance index, initial conditions and simulation process are almost the same as for the 2P3E scenario except that there is no E3 involved in the state equations and performance index. So the state equations are equations (3.9) and (3.10). The controls are the same as the 2P3E scenario as seen in equation (3.12). The performance index is

$$\begin{aligned} J = w_1 \cdot t_f + w_2 \cdot \int_0^{t_f} (\dot{x}_{p1}^2 + \dot{y}_{p1}^2 + \dot{x}_{p2}^2 + \dot{y}_{p2}^2) dt \\ + w_3 \cdot (x_{e1}^2(t_f) + y_{e1}^2(t_f) + x_{e2}^2(t_f) + y_{e2}^2(t_f)) \end{aligned} \tag{3.14}$$

Since E3 is only involved in the simulation part and is reacting according to the motions of both pursuers, the simulation process is the same as the 2P3E scenario.

2. Results of Designed and Simulated Trajectories

In this scenario, the shapes of designed trajectories in Figure 25 and the corresponding controls, relative distance, and relative angle time histories in Figure 26 look very similar to the 2P3E scenario even though E3 has no effect on the designed motion of the pursuers. When $w_1 = 1$ and $w_2 = 1$ (time and effort optimal case), the results are $t_f = 8.9192$ seconds and $J = 22.5921$ with the weight of the soft constraint set to $w_3 = 100$. One can see that t_f is lower and J is higher than the two sets of independent 1P1E scenario.

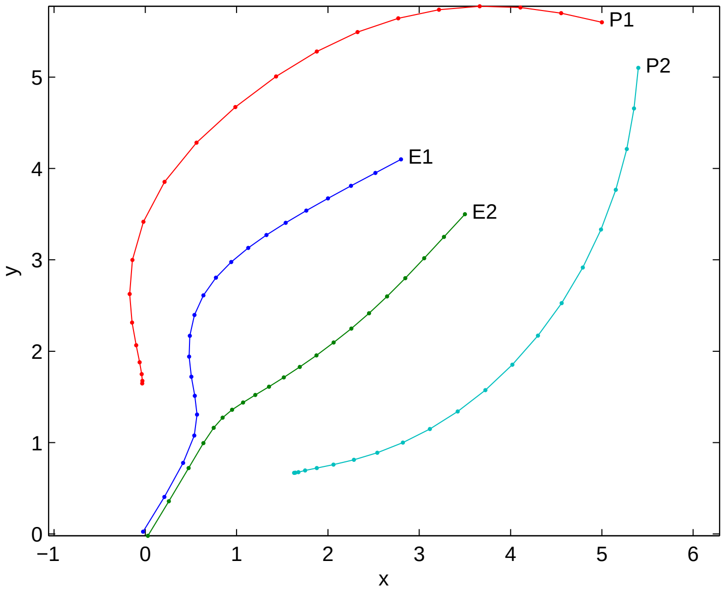

Fig. 25. Designed Trajectories for 2P2E and Time and Effort Optimal

After simulating the trajectories with different initial positions of E3, Figure 27 shows that E3 does not always approach the goal, but moves away from the goal when the initial position (in this case $[x_{e3}(t_0), y_{e3}(t_0)] = [3.3, 3.9]$) is a little further from the goal that other two evaders.

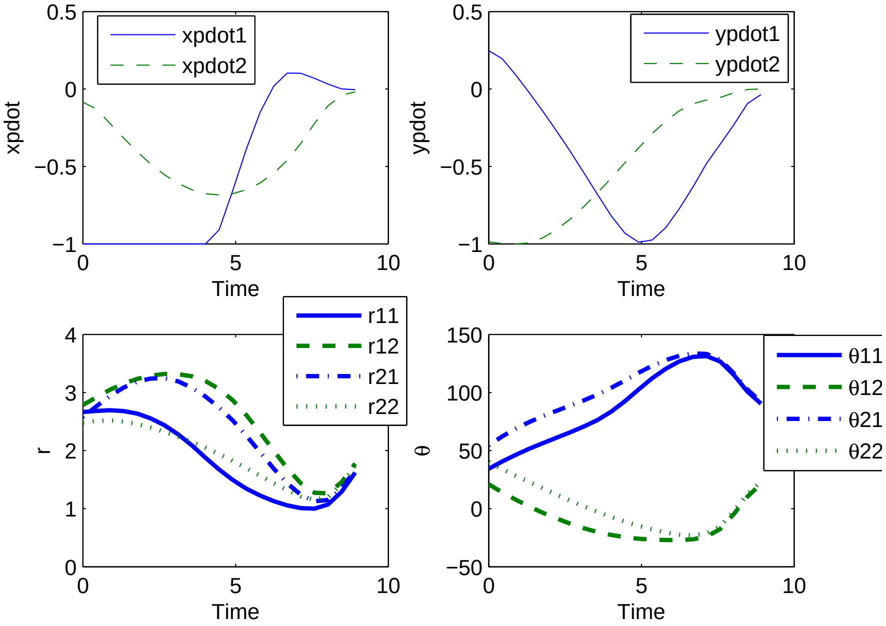

Fig. 26. Control and Relative Vector Time Histories for 2P2E and Time and Effort Optimal

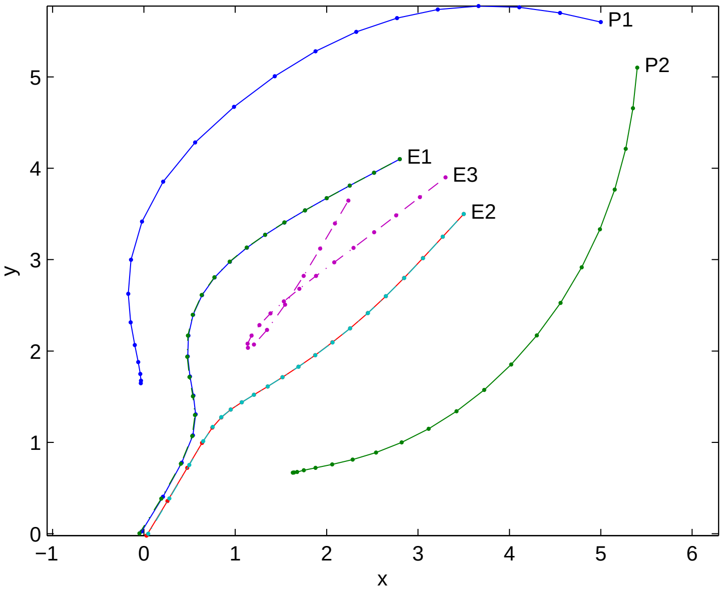

Fig. 27. Designed and Simulated Trajectories for 2P2E+E3 and Time and Effort Optimal

3. Conclusion and Discussion

Following the 2P3E scenario, we removed the soft constraint, $x_{e3}^2(t_f) + y_{e3}^2(t_f)$, from the performance index in order to compare the resulting performance index to the 2×1P1E scenario. The results show that the final time as a part of performance index is improved by using cooperation as opposed to the 2×1P1E scenario for the time and effort optimal case because the influences from each pursuer have a larger net effect on the evaders. However, J (excluding the contribution of the soft constraint term in the performance index) is increased since cooperation of pursuers requires more control effort. Additionally, in the simulation part, E3 does not always approach the goal when the soft constraint is not included in the performance index because the pursuers do not take the trajectory of E3 into account. If we examine state equation (3.11), the right side of the equations will become positive if the motion of E3 is slower than two pursuers, and they pass it. This occurs when E3 is a little further from the goal that other evaders as shown in Figure 27. This causes E3 to move in the opposite direction, away from the goal. In contrast, if E3 is included in the designed trajectories as in the 2P3E scenario, it shows that E3 will approach the goal as seen in Figure 20.

D. Two Pursuers and Four Evaders Scenario (2P4E)

To show the further benefits of cooperation, we include more cheap “dumb” robots in our problem. Increasing the number of evaders is very challenging as it increases the difficulty of numerical computation. In this section we study the ability of the methodology to handle more evaders.

1. System Model, Formulation of Optimal Control and Simulation Process

This scenario is very similar to the 2P3E scenario. We start with 2P3E scenario and add one more evader (E4) to the state equations, performance index and simulation process. Figure 28 shows that the model for each individual evader. The figure shows that any single evader (represented by i) has two relative distance vectors relating two pursuers as well as two relative position angles. Thus, in addition to the state equations (3.9), (3.10) and (3.11) in the 2P3E scenario, we include state equation (3.15) for E4:

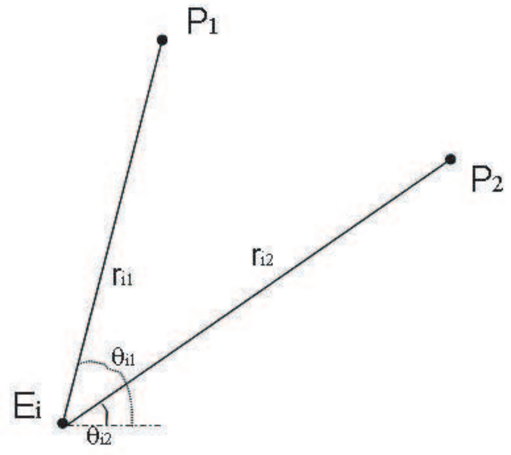

Fig. 28. System Model for Each Single Evader

$$\mathrm{E}_4 : \begin{cases} \dot{x}_{e4} &= -\frac{1}{r_{41}}\cos(\theta_{41}) - \frac{1}{r_{42}}\cos(\theta_{42}) \\ \dot{y}_{e4} &= -\frac{1}{r_{41}}\sin(\theta_{41}) - \frac{1}{r_{42}}\sin(\theta_{42}) \end{cases} \tag{3.15}$$

$$\begin{aligned} \text{where: } r_{41} &= \sqrt{(x_{p1} - x_{e4})^2 + (y_{p1} - y_{e4})^2} \\ \theta_{41} &= \tan^{-1}\left(\frac{y_{p1} - y_{e4}}{x_{p1} - x_{e4}}\right) \end{aligned}$$

$$\begin{aligned} r_{42} &= \sqrt{(x_{p2} - x_{e4})^2 + (y_{p2} - y_{e4})^2} \\ \theta_{42} &= \tan^{-1}\left(\frac{y_{p2} - y_{e4}}{x_{p2} - x_{e4}}\right) \end{aligned}$$

The controls are the same as the 2P3E scenario as seen in equation (3.12). The performance index with E4 included is:

$$\begin{aligned} J = w_1 \cdot t_f + w_2 \cdot \int_0^{t_f} (\dot{x}_{p1}^2 + \dot{y}_{p1}^2 + \dot{x}_{p2}^2 + \dot{y}_{p2}^2)dt + w_3 \cdot (x_{e1}^2(t_f) + y_{e1}^2(t_f) \\ + x_{e2}^2(t_f) + y_{e2}^2(t_f) + x_{e3}^2(t_f) + y_{e3}^2(t_f) + x_{e4}^2(t_f) + y_{e4}^2(t_f)) \end{aligned} \quad (3.16)$$

In addition to the initial conditions in 2P3E scenario, we include $[x_{e4}(t_0), y_{e4}(t_0)] = [2.5, 3.4]$ (chosen randomly). The simulation process is as following:

- Initial Condition: $[x_{e1}(t_0), y_{e1}(t_0)]$, $[x_{e2}(t_0), y_{e2}(t_0)]$, $[x_{e3}(t_0), y_{e3}(t_0)]$ and $[x_{e4}(t_0), y_{e4}(t_0)]$
- Given: Same as the 2P3E scenario
- Solve: State Equations (3.9), (3.10), (3.11) and (3.15) for E4

2. Results of Designed and Simulated Trajectories

For convenience, we summarize the results in Table II.

Table II. Results for 2P4E scenario

	Time and Effort Optimal	Effort Optimal	Time Optimal
t_f	10.3674	11.7562	8.6942
J	28.4106	17.2611	14.2778
Computation Time	24.4365	26.0727	49.5807

The results are very similar to the 2P3E scenario. They demonstrate that two pursuers are able to successfully drive a team of four evaders to the goal for three cases. The simulated and designed offline trajectories are plotted in the same figure and they match well, as shown in Figure 29, 30 and 31. The corresponding controls and relative distance and angle time histories can be seen in Figure 32, 33 and 34. We can see that the plots for the time optimal case still display chatter.

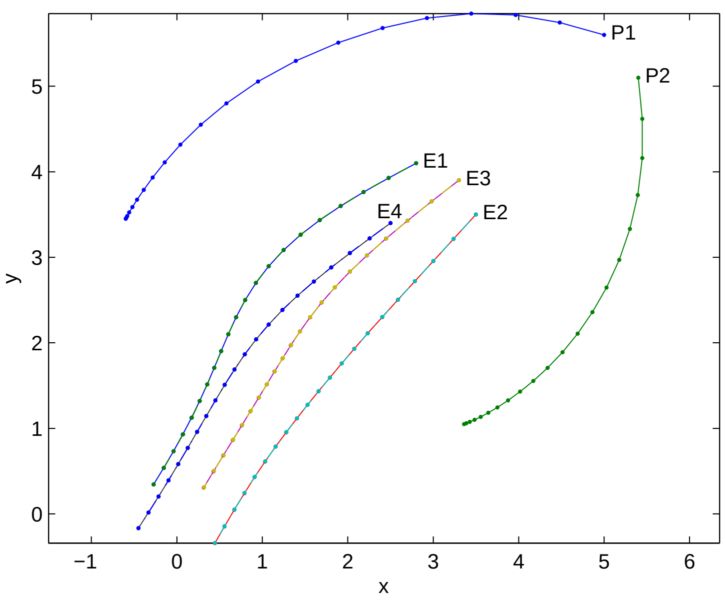

Fig. 29. Designed and Simulated Trajectories for 2P4E and Time and Effort Optimal

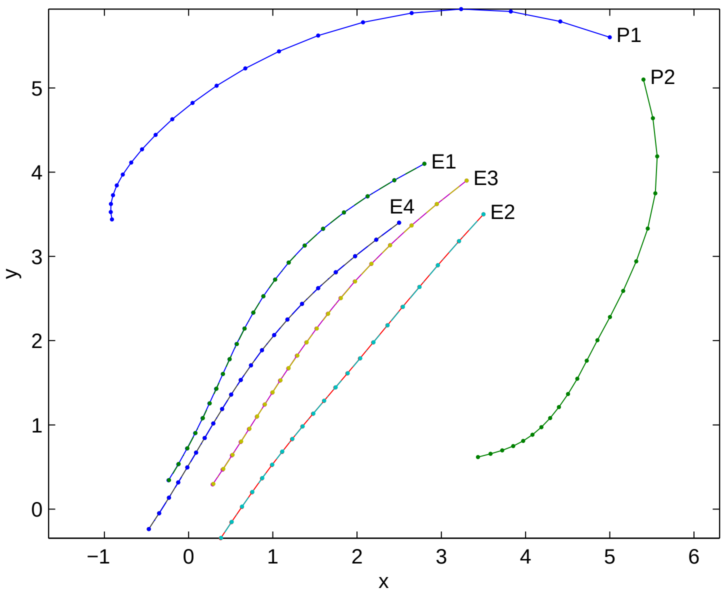

Fig. 30. Designed and Simulated Trajectories for 2P4E and Effort Optimal

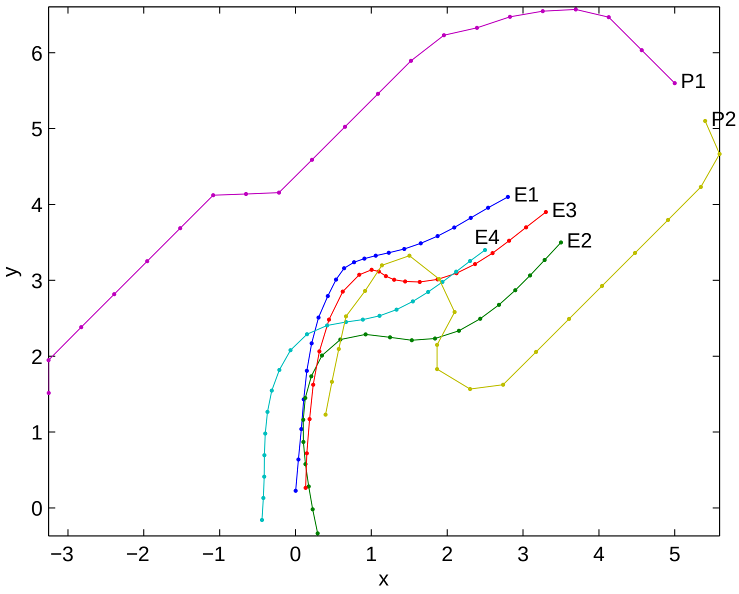

Fig. 31. Designed and Simulated Trajectories for 2P4E and Time Optimal

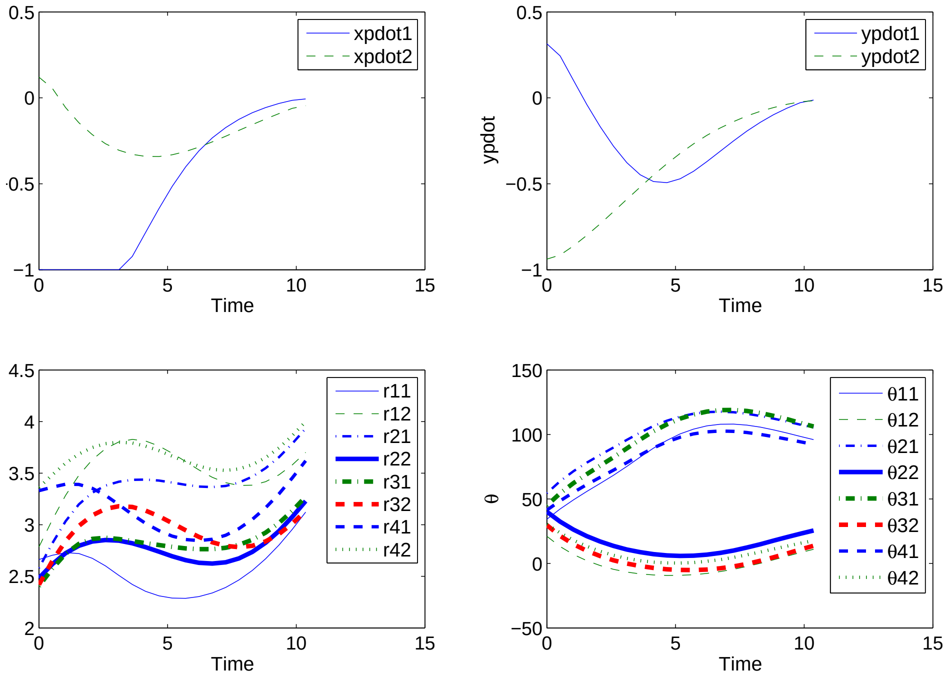

Fig. 32. Control and Relative Vector Time Histories for 2P4E and Time and Effort Optimal

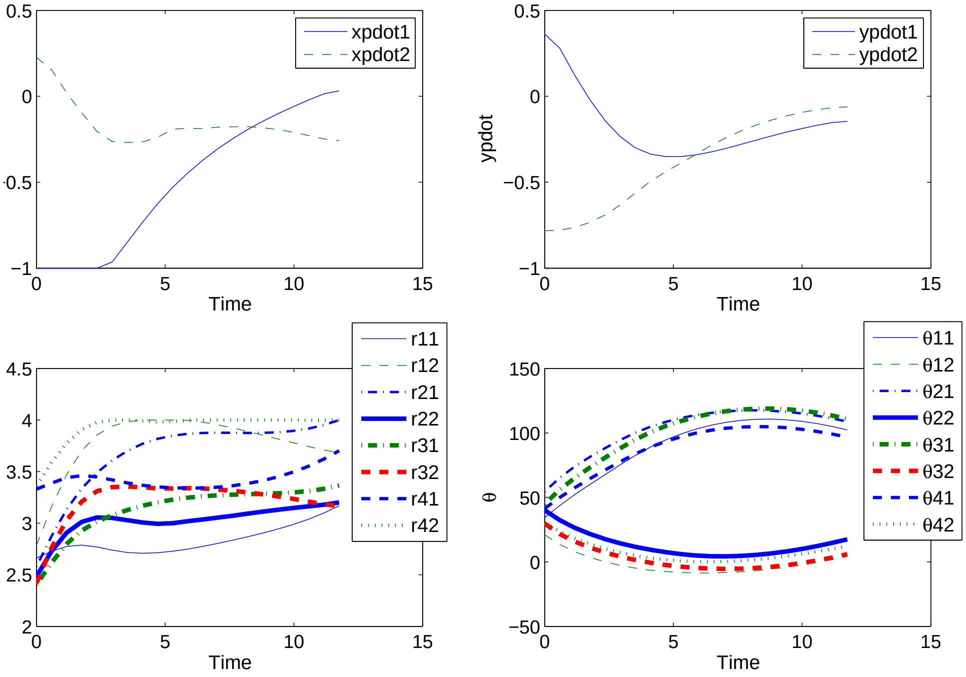

Fig. 33. Control and Relative Vector Time Histories for 2P4E and Effort Optimal

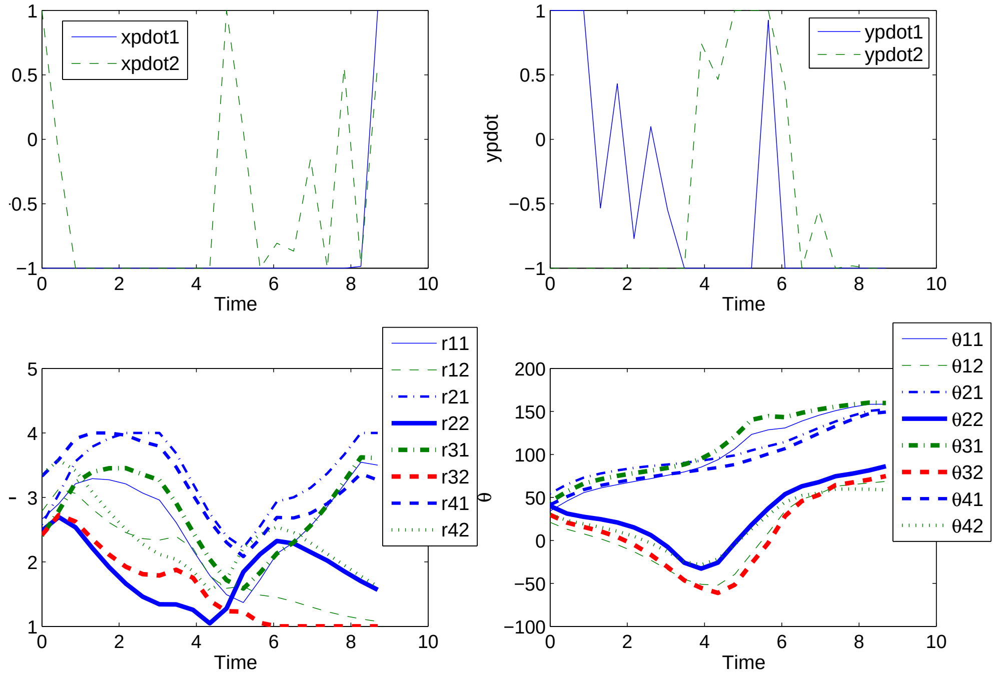

Fig. 34. Control and Relative Vector Time Histories for 2P4E and Time Optimal

In the time optimal case shown in Figure 31, P1 seems to move away from the evaders toward the end of the simulation. In order to get a clearer picture of the time optimal case, we include three sub-cases, created by changing the weight of soft constraints, w_3 in the performance index. Figure 35, 36 and 37 show the trajectories with the weight w_3 =1, 5 and 100 respectively. Three figures show that when increasing w_3, P1 has higher tendency to move further from the evaders in order to maneuver into a better position to herd four evaders. When $w_3 = 100$, P1 eventually goes around and passes the evader to push them back toward [0,0].

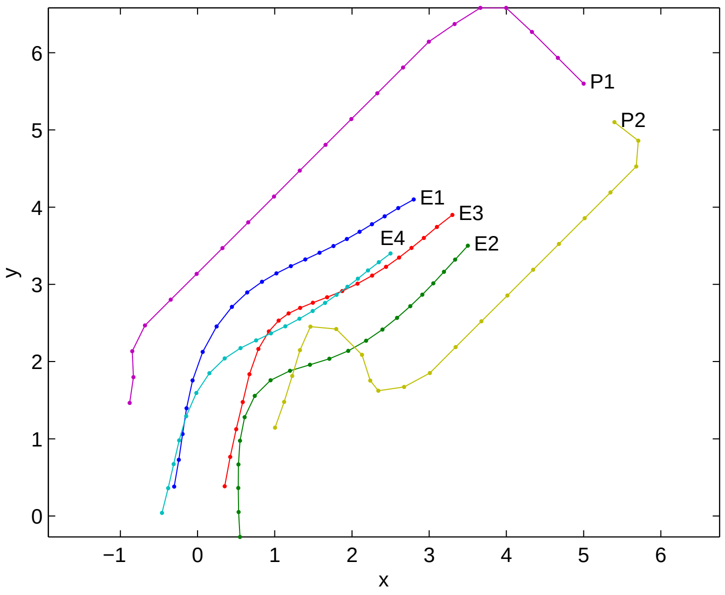

Fig. 35. Designed and Simulated Trajectories for 2P4E and Time Optimal ($w_3 = 1$)

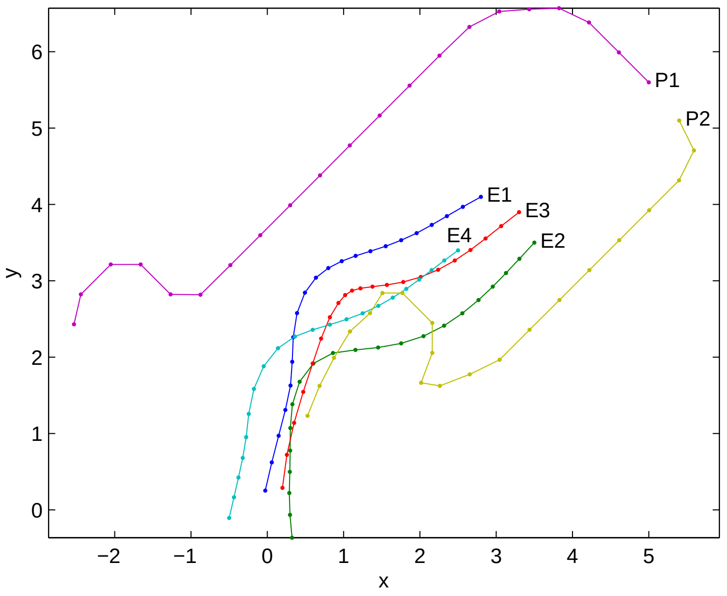

Fig. 36. Designed and Simulated Trajectories for 2P4E and Time Optimal ($w_3 = 5$)

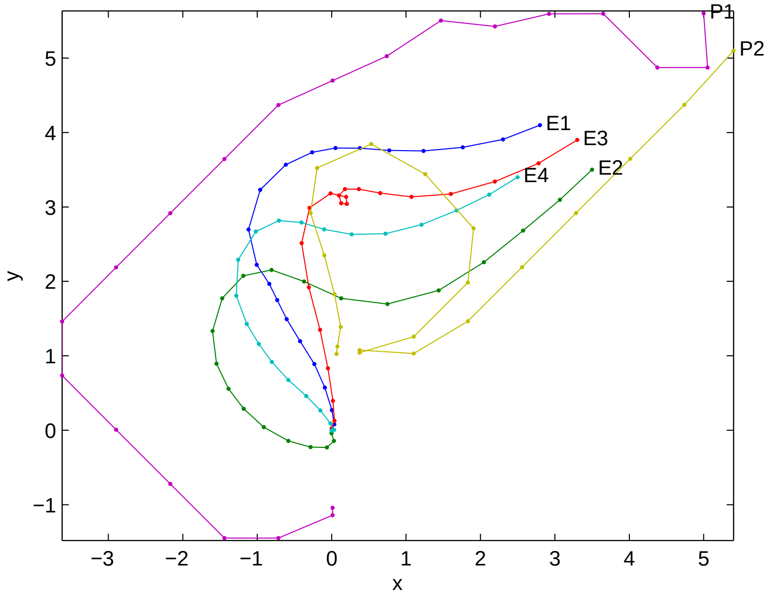

Fig. 37. Designed and Simulated Trajectories for 2P4E and Time Optimal ($w_3 = 100$)

3. Conclusion and Discussion

Following the 2P3E scenario, we studied the 2P4E scenario in this section as an extension. We use the same the process as was used previously except for the addition of adding E4 in the state equations, performance index, initial conditions and simulation process. The results show that two pursuers are able to successfully drive a team of four evaders to the goal for three cases. The plots look very similar to those of the 2P3E scenario. The final time, t_f, is minimum in the time optimal case and maximum in the effort optimal case. The performance index J is highest in the time and effort optimal case and lowest in the time optimal case as we expect. However, the time optimal case is most numerically challenging case. The simulated trajectories match those designed offline very well. The computation time dose not increase much compared to the 2P3E scenario despite having 20×N+9 constraint functions. This means that the cooperative model can be applied to the four evaders scenario, and maybe more as long as the computation limit allows. The process of adding one evader (denoted by Ei) to the 2P4E scenario is the same as the process followed to go from the 2P3E scenario to the 2P4E scenario.

CHAPTER IV

CONCLUSION AND DISCUSSION

In this thesis, we study a new type of pursuit-evasion game, which we call the herding problem. Unlike typical pursuit evasion games, where the pursuer aims to catch or intercept the evader, the goal of the pursuers in this game is to drive the evaders to a certain location or region in the x-y plane. Starting from the one pursuer and one evader scenario, a mathematical model was developed that governs the motion of the evader away from the pursuer. Using this model, we formulated a basic herding problem as an optimal control problem by defining state equations, controls, constraints and a performance index and then focused on computing solutions numerically through dynamic programming by utilizing the SNOPT software in the MATLAB environment. The solution process includes using random initial conditions, solving the designed offline trajectories and then simulating the motion of the system. Once the 1P1E scenario was well established, a cooperation model was developed in a two pursuer and two or multiple evaders scenario. Then we developed the formulation of optimal control problems and solved them numerically as we had done in the 1P1E scenario. The results showed that we were able to simulate two pursuers driving a team of three evaders, instead of just two, successfully to the goal for three cases, created by changing the weights in the performance index. Additionally, we noticed that when t_f is included as a part of performance index the final time is improved with cooperation compared to the two sets of independent 1P1E scenario. Also, from the results, we observed that the time duration of the herding, t_f, was the lowest for the time optimal case and the highest for the effort optimal case. The performance index J was the highest in the time and effort optimal case and the lowest in the effort optimal case. When simulating the response of the system, despite the difference in integration method, the designed offline and simulated trajectories are very consistent with each other in the 2P3E scenario and match fairly well in the 1P1E scenario. Inevitably, there are always numerical errors in the time optimal case and the plots display control chatter because we penalized only the unfixed t_f. However, there are methods to smooth chattering controls that may alleviate this problem. In summary, this research can be implemented in experiments easily after all the steps are established. Finally, one may consider the very natural and interesting extensions of problems of this cooperative model to this kind of scenario. If we have a group of multiple expensive robots, which we call pursuers (more than two) and a group of multiple cheap robots, which we call evaders, we can apply this cooperative model by dividing a group of multiple pursuers into different teams. Each team has two pursuers or one (if the number of pursuers is odd). Then each team can herd 3 or 4 evaders to the goal. A methodology for dividing pursuers into teams and a process of choosing which evaders a team should herd is a topic of future research.

The main contribution of this thesis are the direction and speed evasion model and the influence vector policy that allows cooperation among the players.

REFERENCES

[1] J. R. Krebs and N. B. Davies, *Behavioural Ecology: An Evolutionary Approach*, pp. 169–196. Oxford: Blackwell Publishing, 1991.

[2] R. Isaacs, *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization.* New York: John Wiley and Sons, Inc., 1965.

[3] A. E. Bryson and Y. C. Ho, *Applied Optimal Control: Optimization, Estimation, and Control*, pp. 271–295. Waltham, MA: Blaisdell Publishing, 1969.

[4] S. Murtaugh and H. Criel, "Fundamentals of proportional navigation," *IEEE Spectrum*, vol. 3, pp. 75–85, December 1966.

[5] M. Guelman, "A qualitative study of proportional navigation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-7, pp. 637–643, July 1971.

[6] M. Guelman, "Proportional navigation with a maneuvering target," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-8, pp. 364–371, May 1972.

[7] A. E. Puckett and S. Ramo, *Guided Missile Engineering*, pp. 147–185. York, PA: McGraw-Hill, 1959.

[8] R. Behn and Y. C. Ho, "On a class of linear stochastic differential games," *IEEE Transactions on Automatic Control*, vol. 13, pp. 227–240, June 1968.

[9] S. A. Shedied, *Optimal Control for a Two Player Dynamic Pursuit Evasion Game; The Herding Problem.* PhD thesis, Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, January 2002.

[10] J.-M. Lien, O. B. Bayazit, R. T. Sowell, S. Rodriguez, and N. M. Amato, “Shepherding behaviors,” in *Proc. of the IEEE Int. Conf. Robot. Autom.*, (New Orleans, LA), pp. 4159–4164, April 2004.

[11] J.-M. Lien, S. Rodriguez, J.-P. Malric, and N. M. Amato, “Shepherding behaviors with multiple shepherds,” in *Proc. of the IEEE Int. Conf. Robot. Autom.*, (Barcelona, Spain), pp. 3402–3407, April 2005.

[12] N.-S. Kang, “Cooperative optimal path planning with functional surface based on the finite element method,” Master’s thesis, Texas A&M University, College Station, Texas, May 2002.

[13] J. Feddema, C. Lewis, and D. Schoenwald, “Decentralized control of cooperative robotic vehicles: Theory and application,” *IEEE Transactions on Automatic Control*, vol. 18, pp. 851–864, October 2002.

[14] J. Hurtado and R. R. III, “Decentralized control for a swarm of vehicles performing source localization,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 41, pp. 1–18, 2004.

[15] J. Hurtado and R. R. III, “Cooperative localization and tracking of time-dependent sources,” in *Proc. of the American Nuclear Society 10th International Conference on Robotics and Remote Systems for Hazardous Environments*, (Gainesville, FL), pp. 115–122, March 2004.

[16] P. E. GILL, *Users Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming.* Department of Mathematics, University of California, San Diego, La Jolla, CA 92093-0112, September 2005.

VITA

Zhenyu Lu received his Master of Engineering degree in Civil Engineering at Texas A&M University in August, 2004 and received his Bachelor of Engineering degree in Civil Engineering at Tongji University in Shanghai, China, in June, 2002. He studied for his Master's degree in Aerospace Engineering at Texas A&M University under Dr. John E. Hurtado. He received his Master of Science degree in December 2006. His major specialization is dynamics and controls in the field of Aerospace Engineering. His permanent address is 99 East Guo Quan Road No. 30 Room 1101, Shanghai, China 200433.

The typist for this thesis was Zhenyu Lu.