2013

Bio-inspired Cooperative Optimal Trajectory Planning For Autonomous Vehicles

Charles Remeikas
University of Central Florida

BIO-INSPIRED COOPERATIVE OPTIMAL TRAJECTORY PLANNING FOR AUTONOMOUS VEHICLES

by

CHARLES REMEIKAS
B.S. University of Central Florida, 2011

A thesis submitted in partial fulfillment of the requirements
for the degree of Masters of Science
in the Department of Mechanical & Aerospace Engineering
in the College of Engineering & Computer Science
at the University of Central Florida
Orlando, Florida

Fall Term
2013

ABSTRACT

With the recent trend for systems to be more and more autonomous, there is a growing need for cooperative trajectory planning. Applications that can be considered as cooperative systems such as surveying, formation flight, and traffic control need a method that can rapidly produce trajectories while considering all of the constraints on the system. Currently most of the existing methods to handle cooperative control are based around either simple dynamics and/or on the assumption that all vehicles have homogeneous properties. In reality, typical autonomous systems will have heterogeneous, nonlinear dynamics while also being subject to extreme constraints on certain state and control variables. In this thesis, a new approach to the cooperative control problem is presented based on the bio-inspired motion strategy known as local pursuit. In this framework, decision making about the group trajectory and formation are handled at a cooperative level while individual trajectory planning is considered in a local sense. An example is presented for a case of an autonomous farming system (e.g. scouting) utilizing nonlinear vehicles to cooperatively accomplish various farming task with minimal energy consumption or minimum time. The decision making and trajectory generation is handled very quickly while being able to consider changing environments laden with obstacles.

ACKNOWLEDGMENTS

First and foremost I would like to thanks my advisor Dr. Yunjun Xu for his guidance, patience, and support through my academic career. Without his direction and help I would not be where I am at today.

I would also like my committee members Dr. Kuo-chi Lin and Dr. Alain Kassab for accepting the responsibility of helping with the defense of my thesis.

Many of the people in my lab have been a great asset in accomplishing my research and academic goals including Dr. Gareth Basset, Robert Sivilli, Ni Li, Jacob Belli, Brad Sease, and He Shen. It has been an honor to learn from and work with these individuals.

Specifically I would like to thanks my wife Jessica Remeikas for her constant love, support, and encouragement in the last 7 years of my academic career. She has truly made it possible for me to reach my goals.

Finally, I like to thank my family, especially my parents Teresa Mize and Joseph Remeikas, for their love and support through my life allowing me to be in the position to pursue whatever path I chose.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE: INTRODUCTION

Motivation

Trajectory planning for autonomous systems has been a very hot topic in research in the last decade. It is important to be able to rapidly generate these trajectories if they are going to be used in a system implemented in real time. There are many challenges associated with generating trajectories for vehicles that have nonlinear dynamics that may also be subject to many different severe and realistic constraints. Some examples of these constraints could be obstacle avoidance and strict control limitations. Finding an optimal trajectory for vehicles is also important for many applications. If the system is sufficiently large, or the cost of operating the system is very high, finding an optimal solution could be vital to the success or failure of a system. Incorporating this type of trajectory generation to an application that requires vehicles to operate cooperatively results in a very complex and difficult problem to solve.

The need for a method to generate cooperative trajectories is apparent in many different areas. These range from agricultural purposes like planting and harvesting to military defense issues including surveillance and battle field scouting (Murray 2007). These types of systems require the vehicles to be able to respond quickly, handle changing and cluttered environments, and also consider realistic dynamics. Many of the current algorithms and methods to handle these problem suffer from issues related to broad assumptions or simplifications of the dynamics of the systems (Kim and Meshbai 2006, Wang and Xin 2010) . Since a system of this nature may require the use of a large number of vehicles, the algorithm must be heavily scalable to keep the computational cost low.

Many studies have been conducted on various types of formation control algorithms. In (Murray 2006) many different applications are suggested that can be viewed as a cooperative control problem. In this paper three major types of solutions are reviewed: optimization, potential field solution, and swarm approaches. Optimization based methods, which will be studied in this paper, utilize optimization of various performance indices that relate the followers to one another in a cooperative sense (Dunbar and Murray 2004). These types of methods are typically high in computation cost due to the requirement of solving an optimal control problem. Potential field solutions (Leonard and Fiorelli 2001) use a potential function that is based on the system dynamics, and in this approach these functional are utilized in a leader follower structure to help attract or repel the followers to a leader. In swarming type approaches the vehicles are all moving in some cooperative direction but may not have individual requirements on formation (Reynolds 1987). This type of method is similar to the gradient approaches in that there is an overall goal and the followers are attracted or repelled from each of the other vehicles in the system.

In (Gou et al. 2010) an adaptive leader-follower method is proposed for the formation control of multiple vehicles. The method presented utilizes graph theories to help distribute the follower robots into the desired formation. Control laws for the leaders and followers are derived that allow formation control but has restrictions on the shape and number of followers in a specific formation. In (Lawton et al. 2003) three decentralized methods to handle formation control are presented. These methods all involve a lot of information communication between the individual vehicle and some of the neighboring vehicles. Results are shown that formation control is achieved in actual robotic hardware test.

In this thesis, a method to rapidly generate cooperative trajectories for systems of vehicles with heterogeneous dynamics considering obstacle avoidance and severe constraints is studied. Using a hierarchical leader-follower structure and a bio-inspired modified local pursuit strategy, these challenges can be met. Often animals in nature utilize simple rules to help plan and guide their motions in foraging, mating, and hunting. Local pursuit, a method derived from the movement of ants, is one of many different bio-inspired trajectory generation strategies. Using this motion strategy, each of the followers can calculate their own trajectories based on the relationship between the leader and follower. This allows for the overall computational cost to be reduced.

Two major investigations are conducted, the viability and design of a framework to be used on a system of vehicles, and applying this framework to a realistic application in a citrus harvesting example.

Contributions of Thesis

The contributions of this thesis are that a framework for rapidly generating cooperative optimal trajectories for a group of nonlinear vehicles is designed. This method is studied extensively to consider various aspects that could be vital to implementing into a real system such as the communication complexity, problem dimension and optimality. Monte-Carlo simulations are presented to show the effectiveness and computation cost of the algorithm for three and five follower cases.

The framework is also applied to a realistic citrus harvesting example on a group of cooperative heterogeneous vehicles. In addition to the trajectory generation, a cooperative

decision making structure is also used to decide which configuration the group will use in a given task planning phase.

The benefits of this method shown in (Xu, Remeikas, and Pham 2011) are as follows, (1) the formation of vehicles can be globally, asymptotically maintained, (2) the trajectory of each vehicle can be generated using only its local information and the information of the virtual leader, (3) the convergence speed and optimization of the local trajectories is based only on a single parameter, (4) nonlinear heterogeneous dynamics with realistic constraints are considered, and (5) the computational cost of generating the trajectories for a large group of vehicles is low.

Thesis Outline

The thesis will be organized as follows; First, in Chapter 2, the problem definition about the hierarchal leader follower framework is discussed. Chapter 3 introduces the overall iterative hierarchical architecture and the modified local pursuit based cooperative trajectory planning algorithm are introduced and analyzed. Chapter 4 provides simulation examples to demonstrate the capabilities of the new algorithm including applications to a group of autonomous farming vehicles. Finally, conclusions are given in Chapter 5.

CHAPTER TWO: PROBLEM FORMULATION

A leader-follower structure is adopted in the cooperative system. Here the leader can be a virtual one, e.g. the centroid of the formation, and is denoted as the VL.

The performance indices and constraints, considered in the nonlinear constrained cooperative trajectory planning problem, are grouped into either the cooperative level behavior or the local level behavior categories.

Cooperative Level Behaviors

The following behaviors are considered in the cooperative level (Xu, Remeikas, and Pham 2011)

(i) The communication complexity, i.e. the total information transmitted between the VL and the followers, needs to be low. Communication among followers should be minimized or avoided if the communication between the VL and the followers is available.

(ii) The trajectory of the *VL*, $\boldsymbol{x}_{VL,p} \in \Re^{n_p}$, needs to be rapidly computed and the overall performance of the formation

$$J = J(\boldsymbol{X}, \boldsymbol{U}, t_f) = \int_{t_0}^{t_f} \Upsilon(\boldsymbol{X}, \boldsymbol{U}) dt \tag{1}$$

is minimized. The subscript " p " denotes the position state of the vehicles in the formation, and t_0 and t_f are the initial and final time of the planning horizon, respectively.

$\boldsymbol{X}^T = [\boldsymbol{x}_1^T, \boldsymbol{x}_2^T, ..., \boldsymbol{x}_{n_v}^T]$ and $\boldsymbol{U}^T = [\boldsymbol{u}_1^T, \boldsymbol{u}_2^T, ..., \boldsymbol{u}_{n_v}^T]$ are respectively the aggregate state and control vectors of the n_v followers in the formation. In this paper, the total energy index

$$J = \sum_{i=1}^{n_v} \left(\int_{t_0}^{t_f} \boldsymbol{u}_i^T \boldsymbol{u}_i dt \right) = \int_{t_0}^{t_f} \left(\sum_{i=1}^{n_v} \boldsymbol{u}_i^T \boldsymbol{u}_i \right) dt \tag{2}$$

is studied, in which $\Upsilon(\boldsymbol{X}, \boldsymbol{U}) = \sum_{i=1}^{n_v} \boldsymbol{u}_i^T \boldsymbol{u}_i$. It is worth mentioning that different performance indices, such as the minimum time $J = \int_{t_0}^{t_f} dt$ and the minimum tracking error, can also be considered.

(iii) The *VL* may need to satisfy the speed constraint $0 \le V_{VL} \le V_{VL,\max}$, the angular velocity constraint $0 \le \omega_{VL} \le \omega_{VL,\max}$, and the collision avoidance constraint

$$\left\| \boldsymbol{x}_{VL,p} - \boldsymbol{x}_{obs,j} \right\| \ge r_{obs,j} + \Delta_{\max}, \; j = 1, \ldots n_{obs} \tag{3}$$

in which $\boldsymbol{x}_{obs,j}$ and $r_{obs,j}$ are the center and radius of obstacle j, respectively. $\Delta_{\max}$ is the maximum distance bias of the followers with respect to the *VL*. n_{obs} is the number of obstacles observed in the $[t_0, t_f]$ planning horizon.

(iv) Inter-vehicle collision should be avoided, i.e.

$$\left\| \boldsymbol{x}_{i,p} - \boldsymbol{x}_{j,p} \right\| \ge \varepsilon_s, i, j = 1, \ldots, n_v, i \ne j \tag{4}$$

, in which ε_s is the safe distance buffer among different vehicles. It is worth noting that initially the vehicles in the formation should not collide with each other.

(v) The *VL* needs to satisfy equality constraints (E.C.s.) such as the initial and final positions $\boldsymbol{x}_{VL,p}(t_0) = \boldsymbol{x}_{VL,p,0}$ and $\boldsymbol{x}_{VL,p}(t_f) = \boldsymbol{x}_{VL,p,f}$.

The inequality constraints (I.E.C.s) discussed in (iii) and (iv) can be organized as

$$\boldsymbol{g}_{VL}(\boldsymbol{x}_{VL}, \boldsymbol{u}_{VL}, \boldsymbol{X}, t) \le 0 \tag{5}$$

while the E.C.s in (v) can be organized as

$$\boldsymbol{h}_{VL}(\boldsymbol{x}_{VL}, \boldsymbol{u}_{VL}, t) = 0 \tag{6}$$

It is worth noting that since the leader may be a virtual one (with virtual state and control variables $\boldsymbol{x}_{VL}$ and $\boldsymbol{u}_{VL}$), there is no dynamics involved in the *VL*.

In the decision making portion of the cooperative level, the group cooperatively decides on a formation configuration from the set $E_{fc} = \{E_{fc,1}, E_{fc,2}, ..., E_{fc,n_F}\}$ using the following performance index

$$\begin{aligned} J_{k,m} &= W_f J^1_{k,m} + W_b J^2_{k,m} \\ &= W_f \left\| \boldsymbol{x}_{L,k}(t_0) - \boldsymbol{x}_{L,k}(t_f) \right\| + W_b \max\{t^t_{1,k,m}, t^t_{2,k,m}, ..., t^t_{n_R,k,m}\}, \\ & k = 1, ..., n_F, m = 1, ..., n_{k,b} \end{aligned} \tag{7}$$

Each of the configuration sets is constructed by $E_{fc,k} = \{\Delta_{k,1}, \Delta_{k,2}, ..., \Delta_{k,n_{k,b}}\}, k = 1, ..., n_F$ where $\Delta_{k,m} \in \Re^{2n_R \times 1}$ is the desired position bias constructing the formation.

Local Level Behaviors

The following behaviors are considered in the trajectory planning of each follower in the formation.

(i) The motion of the i^{th} follower is governed by a class of nonlinear, uncoupled, and possibly heterogeneous dynamics

$$\dot{\boldsymbol{x}}_i = \boldsymbol{f}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, t),\ i = 1, ..., n_v \tag{8}$$

in which n_v is the number of vehicles, and $\boldsymbol{x}_i \in \Re^{n_i}$ and $\boldsymbol{u}_i \in \Re^{m_i}$ are the state and control variables of the i^{th} follower, respectively. $\boldsymbol{x}_{i,p} \in \Re^{n_p}$ denotes the position state in $\boldsymbol{x}_i$, for

example the position of a robot. It is worth noting that in addition to the position state $\boldsymbol{x}_{i,p}$, there may be other state variables in $\boldsymbol{x}_i$, such as the velocity, and these remaining state variables are regarded as the state rate vector $\boldsymbol{x}_{i,sr} \in \Re^{n_i - n_p}$.

Assumption 2: The state and control variables $\boldsymbol{x}_i$ and $\boldsymbol{u}_i$ can be represented using the position state $\boldsymbol{x}_{i,p}$ via the differential inclusion technique or the dynamic inversion procedure (Kumar and Seywald 1996).

Many vehicle models fall under Assumption 2 and example dynamics that satisfy this assumption include but are not limited to: classical DC motor (Guarino Lo Bianco and Piazzi 2002) and spacecraft (Bajodah 2009).

Assumption 3: The system dynamics $\dot{\boldsymbol{x}}_i = \boldsymbol{f}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, t)$ is not under-actuated; otherwise the differential inclusion method in Assumption 2 is not effective, because pseudo-inverse operations may be used and large numerical errors will be generated.

(ii) **(C1)** The followers should avoid collisions with the n_{obs} obstacles as

$$\left\| \boldsymbol{x}_{i,p} - \boldsymbol{x}_{obs,j} \right\| \geq r_{obs,j},\ i = 1,...,n_v,\ j = 1,...n_{obs} \tag{9}$$

Assumption 4: The obstacle information is available to the vehicles in the formation at the beginning of each planning horizon. A planning horizon is predefined or updated when a discrete event is detected. For simplicity, the shape of the obstacles is assumed to be circular as described in Eq.(3) and Eq.(9). Other shapes can be approximated by circles or ellipses, thus are not discussed in this paper.

(iii) **(C2)** The translational and rotational speeds of the followers may be constrained by $0 \le V_i \le V_{i,\max}$ and $0 \le \omega_i \le \omega_{i,\max}$, while the control may be limited by $|\boldsymbol{u}_i| \le \boldsymbol{u}_{i,\max}$.

(iv) **(C3)** Follower i should satisfy boundary conditions, such as $\boldsymbol{x}_{i,p}(t_0) = \boldsymbol{x}_{i,p,0}$ and $\boldsymbol{x}_{i,p}(t_f) = \boldsymbol{x}_{i,p,f}$. Different boundary conditions are considered as to be described in Section 4.B.

The inequality constraints (I.E.C.s) discussed in (ii) and (iii) can be organized as

$$\boldsymbol{g}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, t) \le 0,\ i = 1, ..., n_v \tag{10}$$

, while the E.C.s in (i) and (iv) can be organized as

$$\boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, t) = 0,\ i = 1, ..., n_v \tag{11}$$

(v) The formation of the cooperative followers needs to be maintained and the formation error is defined as

$$\boldsymbol{e}_i = \boldsymbol{x}_{i,p} - \boldsymbol{x}_{VL,p} - \Delta_i, i = 1, ..., n_v \tag{12}$$

, in which Δ_i is the desired formation geometry bias of follower i. The geometry formation should be asymptotically maintained, i.e. as $t \to \infty$, $\boldsymbol{e}_i \to 0$, $i = 1, ..., n_v$, and a certain parameter can be used to conveniently control the formation error decay rate.

(vi) To be consistent with the overall goal in Eq.(1), each follower minimizes its own performance index

$$J_i = J_i(\boldsymbol{x}_i, \boldsymbol{u}_i, t_f) \tag{13}$$

As an example, the minimum energy index is

$$J_i = \int_{t_0}^{t_f} \left(m_i V_i^2 + I_i \delta_i^2 \right) dt \tag{14}$$

While only the energy based performance index is utilized in this paper, many other types could be considered. Some examples are minimal time, minimal distance, or maximum area. The computation cost of the formulated nonlinear programming problem will not be significantly affected by changing the performance index; instead the problem dimension plays a more important factor

Assumption 5: The geometry bias of the follower with respect to the *VL* is not rotating. However the rotation effect can be achieved by reassign the formation geometry bias during trajectory re-planning.

The following coupled constraint is considered in the paper. **(C4)**: the relative distances of the robots are maintained and the conflicts between all of the robots must be resolved as

$$\left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\| \geq \varepsilon_{ij}, i, j = 1, ..., n_R, i \neq j \tag{15}$$

where $\varepsilon_{ij} = \varepsilon_{ji}$ is the safe buffer distance between robot i and robot j considering the sizes of robots.

CHAPTER THREE: BIO-INSPIRED TRAJECTORY OPTIMIZATION METHOD

Decision Making

In the cooperative level as outlined in Algorithm Framework section, a decentralized decision making framework is proposed. In this section, the formation configurations, desired position assignments, and (virtual) leader trajectories are computed and ranked (Xu, Remeikas, and Pham 2011).

Collision Free Trajectory Generation

For each of the formation configuration $E_{fc,k} = \{\Delta_{k,1}, \Delta_{k,2}, ..., \Delta_{k,n_{k,b}}\}, k = 1, ..., n_F$, the leader's trajectory is to be generated that will be used to decide on the best formation configuration for the current farming task. To distribute the computational load, the follower robots in the system will be engaged to calculate the (virtual) leader's trajectory for each of the formation configuration $E_{fc,k} = \{\Delta_{k,1}, \Delta_{k,2}, ..., \Delta_{k,n_{k,b}}\}, k = 1, ..., n_F$. Algorithm 1 here includes two parts.

First, to provide an initial collision free path, a wavefront path planning algorithm (Ozdal and Wong) is used considering only the obstacles and the initial and final position information of the leader . The other constraints in C2-C4 are not addressed at this step. The wavefront planning algorithm is briefly outlined here and the detailed explanation of this method can be found in (Ozdal and Wong).

In the first step, the farming area is divided into $N_{g,x} \times N_{g,y}$ grids. The maximum width and height of the area are denoted as L_x and L_y. The sizes of the grids $S_{g,x} = L_x / N_{g,x}$ and

$S_{g,y} = L_y / N_{g,y}$ can be changed to provide more accurate results. The grid that is covered even by a portion of an obstacle will be assumed to be covered by that obstacle. The grid location of the center of obstacle j, $\boldsymbol{g}_{j,c}$, is found by

$$\boldsymbol{g}_{j,c} = \left(\left\lceil x_{o,j} / S_{g,x} + 1 \right\rceil, \left\lceil y_{o,j} / S_{g,y} + 1 \right\rceil \right), j = 1, ..., n_O \tag{16}$$

in which the value in $\lceil \ \rceil$ is rounded to the nearest integer. The numbers of grids to be occupied by obstacle j along both directions are $n_{j,x} = \lfloor 2a_j / S_{g,x} \rfloor$ and $n_{j,y} = \lfloor 2b_j / S_{g,y} \rfloor$, respectively, in which $\lfloor \ \rfloor$ denotes the ceiling value. Once the grids of all the obstacles are determined, each grid containing an obstacle is assigned a "-1" value. The grid locations of the initial and final positions of the (virtual) leader robot are calculated and represented as the "start" $\boldsymbol{g}_s$ and "goal" $\boldsymbol{g}_f$, respectively. The grids occupied by the initial and final positions of the leader robot are assigned a "2" value. The grids that are neither occupied by obstacles nor the initial and final positions will be assigned a "0" value.

In the next step, the backwards propagation searches the neighboring grids surrounding the current grid position (with an index of $[i_x, i_y]$ and the value associated with this grid is $\boldsymbol{g}_{i_x,i_y}$) starting from the final position. The values assigned to all the neighbors are: the value of the grid will not be changed if the value is not "0". Otherwise, the value of this grid will be $\boldsymbol{g}_{i_x,i_y} + 1$. This procedure continues until it reaches the initial position grid. Once the initial position grid is reached, any grid in the search area with a value of "-1" is set to a value much larger than the value in grid $\boldsymbol{g}_s$.

Finally, the forward propagation is used starting from the initial grid location $\boldsymbol{g}_s$. The values of all its neighbors are compared and the grid with the minimum value will be set as the next grid location. This propagation is continued until the final grid location $\boldsymbol{g}_f$ is reached. Connecting each of the grid locations found in the forward propagation, an obstacle free corridor is generated.

Though the wavefront method provides a set of obstacle free waypoints for the formation to travel through, the path is typically not smooth due to the discretized grids. Therefore an ad-hoc nonlinear programming problem is solved in Algorithm 1-Part 2 to obtain a smooth path for the leader.

In the second part of the collision free trajectory generation **(P3)**, a B-spline curve (Piegl and Tiller 1997) is used parameterize the (virtual) leader's trajectory (the j^{th} direction, the k^{th} formation) with a small number of control points as

$$x_{L,k,j}(t_l) = \sum_{i=0}^{n_{cp}} B_{i,d}(t_l) P_{k,j,i}\ ,\ \ j = 1,2\ ,\ k = 0,...,n_F\ , l = 0,...,N \tag{17}$$

in which a nonlinear rational B-spline curve of degree d is used. $P_{k,j,i}$ and $B_{i,d}(t)$, $i = 0,...,n_{cp}$ are the j^{th} directional of the leader position and the d^{th} degree basis function, respectively. $n_{cp}+1$ is the number of control points in the B-spline representation of the trajectory. The set of optimizable parameters $S_{O,L}$ in this step is

$$S_1^O = \{P_{k,j,i},\ i = 2,...,n_{cp}-2,\ \ j = 1,2\} \tag{18}$$

In this case, the first and last control points are known based on the B-spline property (Piegl and Tiller 1997) and thus don't need to be optimized since the initial and final positions of the leader are known.

An ad-hoc performance index, the first term in Eq. (7), is discretized and used in the optimization as

$$J_{k,m}^{1} = \sum_{l=0}^{n_{IT}-1} \left\| \boldsymbol{x}_{L,k}(t_{l+1}) - \boldsymbol{x}_{L,k}(t_l) \right\| \tag{19}$$

where n_{IT} is the number of interpolation points used to evaluate the (virtual) leader's trajectory represented via the B-spline curve.

Formation and Position Ranking

With the leader trajectory for each of the formation configurations $E_{fc,k}, k = 1,...,n_F$ and the desired position bias $\Delta_{k,m},\ m = 1,...,n_{k,b}$ generated, the options of the formation and position locations for the follower robots within that formation are ranked.

First, the configurations E_{fc} are ranked based off the performance index $J_{k,m}$ in Eq. (7). Since the portion of the performance index $J_{k,m}^{1}$ has been calculated in the smoothing algorithm (Algorithm 1 - Part2), only the transient times for each follower robot $t_{i,k,m}^{t},\ i = 1,...,n_R$, $k = 1,...,n_F$, $m = 1,...,n_{k,b}$ remains to be determined. Each robot calculates the time it will take to travel from its starting position to each of the available positions in the currently selected formation configuration. The minimum possible transient time $t_{i,k,m}^{t}$ can be calculated as

$$t_{i,k,m}^{t} = \left\| \boldsymbol{x}_{i,k,m}^{*} - \boldsymbol{x}_i(t_0) \right\| / V_{i,\max} \tag{20}$$

in which $\boldsymbol{x}^{*}_{i,k,m}$, $\boldsymbol{x}_i(t_0)$, and $V_{i,\max}$ are robot i's desired position (formation i and desired location j), current position, and maximum speed, respectively.

Once all of the transient times have been determined, the finite dimension problem (Cormen et al. 2009) is solved for the combined performance index in Eq. (7). The number of possible options depends on the number of formation configurations, follower robots, and positions available in the formation.

Local Pursuit Trajectory Generation

In the cooperative foraging motion via the local pursuit (LP) strategy (Fig. 1), an ant may point its velocity towards the position of the ant ahead of it to achieve the minimum time performance (Hristu-Varsakelis and Shao 2004). The simple LP rule is

$$\dot{\boldsymbol{x}}_{i,p}(t) = v_i\left[\boldsymbol{x}_{VL,p}(t) - \boldsymbol{x}_{i,p}(t)\right], i = 1,\ldots,n_v \tag{21}$$

in which $v_i(t)$ is defined as the "speed control parameter" (SCP) and relates to the speed of the follower. To maintain a desired formation, the LP strategy Eq.(21) is modified in this paper as

$$\dot{\boldsymbol{x}}_{i,p} = v_i(\boldsymbol{x}_{VL,p} - \boldsymbol{x}_{i,p}) + v_i\boldsymbol{\Delta}_i + \dot{\boldsymbol{x}}_{VL,p},\ i = 1,\ldots,n_v \tag{22}$$

in which $\boldsymbol{\Delta}_i \in \Re^{n_p}$ is the constant bias in the formation between the *VL* and follower i.

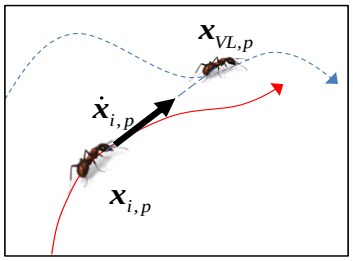


Figure 1 Local pursuit phenomenon found in ants' foraging.

The local level behavior (v) is automatically satisfied using Theorem 1 described next.

Theorem 1: The formation is globally asymptotically stable if the SCP variable of each follower, whose motion is controlled by the modified LP strategy Eq.(22), is positive. The proof of this theorem can be found in (Xu, Remeikas and Pham 2011)

Thus the formation can be maintained globally asymptotically stable if the SCP is kept locally positive. To ensure the SCP stays positive, it is considered as an inequality constraint in the local level optimization. Obstacle avoidance is also taken into account as an inequality constraint in Eq. (3)

It is worth noting that delay can be considered in Eq. (22), although it will not be discussed in this paper. Different values $\Delta_i \in \Re^{n_p}$ can be used for different application scenarios. For example, Δ_i can be a line following the *VL* in the case when a group of aircraft is landing sequentially in an airport. Δ_i can be a line parallel to the *VL* in the case when a group of cars driving in a highway.

Definition 1: $\varepsilon_{i,s} > 0, i = 1,...,n_v$ is defined as the radius of a ball around the desired location of a follower in the formation. Within this ball, the followers will be free of any inter-vehicle collision as shown in Fig. 2.

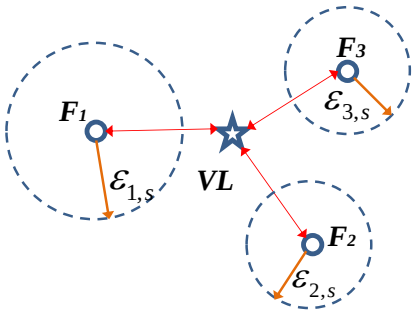


Figure 2 Safe zone defined for a formation.

Corollary 1: If the modified LP strategy Eq. (22) is applied by all the followers, the inter-vehicle collision can be avoided after $t_s = \max(t_{1,s}, t_{2,s}, ..., t_{n_v,s})$, in which

$$t_{i,s} = \begin{cases} -\dfrac{1}{v_{i,\min}} \ln\left(\dfrac{\varepsilon_{i,s}}{\left\| \boldsymbol{e}_{i,p}(t_0) \right\|} \right), & \varepsilon_{i,s} < \left\| \boldsymbol{e}_{i,p}(t_0) \right\| \\ 0, & \varepsilon_{i,s} \geq \left\| \boldsymbol{e}_{i,p}(t_0) \right\| \end{cases}, i = 1, ..., n_v \tag{23}$$

Here $\| \ \|$ is the 2-norm, and $v_{i,\min}$ is the minimum SCP of follower i.

To guarantee the stability of the formation, theorem 1 and corollary 1 only consider first order kinematic relationships. Furthermore, the nonlinear dynamics are considered in the local level optimization where the some state and control variables are solved using differential inclusion or dynamic inversion. Finally, if the nonlinear dynamics do not satisfy the local level LP rule set for the simple first order model, then the optimization iteration will continue.

It is also worth noting that to implement the generated trajectories in the feedback sense, two approaches could be utilized: open-loop planning with a low level tracking controller, or embed the proposed method into a receding horizon framework.

Only the trajectories between the time frame of $[0, t_s]$ need to be transmitted to the *VL* to check the inter-vehicle collision avoidance. However, once the formation is formed, the shape of the geometry is fixed. If obstacles pop up, the whole formation has to either pass through or detour around the obstacles.

Early Termination Conditions

In the nonlinear constrained optimal cooperative trajectory planning algorithm, the early termination conditions are used to reduce the computational cost by not wasting the CPU time in

the local level optimization if these conditions are not satisfied in the cooperative level. If all the early termination conditions are satisfied, the fact that the guessed *VL* trajectory and the formation already satisfy parts or all of the I.E.C.s makes the local level optimization a lot easier.

In addition to the I.E.C.s. described in Eq.(5), another early termination condition that considers the speed limitation of the followers is derived.

Lemma 1: if the speed of follower i at time node k is limited by

$$V_{i,\min} \le V_{i,k} \le V_{i,\max}, k = 0,...,N,\ i = 1,...,n_v \tag{24}$$

, a sufficient condition for the follower to satisfy Eq.(24) is

$$V_{VL,k}^2 - \underline{\lambda}_k \le V_{i,\min}^2, k = 0,...,N,\ i = 1,...,n_v \tag{25}$$

, while a necessary condition is

$$V_{VL,k}^2 - V_{i,\max}^2 < \overline{\lambda}_k, k = 0,...,N,\ i = 1,...,n_v \tag{26}$$

, in which $\overline{\lambda}_k$ and $\underline{\lambda}_k$ are respectively the maximum and minimum eigenvalues of matrix $\boldsymbol{A}_{VL,k} \triangleq \dot{\boldsymbol{x}}_{VL,p,k} \dot{\boldsymbol{x}}_{VL,p,k}^T$.

To obtain the minimum and maximum eigenvalues, the velocity information of the leader is first calculated using the b-spline representation outlined in (Xu, Remeikas, and Pham 2011). The eigenvalues are then calculated using $\boldsymbol{A}_{VL,k} \triangleq \dot{\boldsymbol{x}}_{VL,p,k} \dot{\boldsymbol{x}}_{VL,p,k}^T$.

Equations (5), (32) and (33) are used as the early termination conditions, which can reduce the computational cost in the optimal cooperative trajectory planning.

Based on Lemma 1, the speed of the *VL* must satisfy $V_{VL,k}^2 \le V_{i,\min}^2 + \underline{\lambda}_k$ and $V_{VL,k}^2 < V_{i,\max}^2 + \overline{\lambda}_k$. The first one gives a sufficient solution, i.e., if this condition is satisfied, the speed of follower *i*

must satisfy the speed limitation. The second inequality is a necessary condition, i.e., if there is a solution for follower *i* that satisfies the speed limitation, this condition must be satisfied.

Nonlinear Programming Formulation

Sets $S_{O,i}, i = 1,...,n_v$ and $S_{O,VL}$ include the parameters that are optimized in the local level optimization of follower i and the cooperative level optimization of *VL*. As shown in the following table, the parameters included in $S_{O,i}$ are different for different BCs.

Table 1 Optimizable parameters in the achieved NLPs

Cooperative level		$S_{O,VL} = \{x_{VL,i,j}, i = 1,...,n_p, j = 1,...,n_{cp} - 1\}$
Local level	BC1	$S_{O,i} = \{v_{i,k}, k = 1,...,N\}, i = 1,...,n_v$
	BC2	$S_{O,i} = \{v_{i,k}, k = 1,...,N\}, i = 1,...,n_v$
	BC3	$S_{O,i} = \{v_{i,k}, k = 0,...,N\}, i = 1,...,n_v$

Sets $S_{C,i}$ and $S_{C,VL}$ include the parameters to be calculated, as shown in the following table.

Table 2 Parameters to be calculated in the achieved NLPs

Cooperative level		$S_{C,VL} = \{x_{VL,i,j}, i = 1,...,n_p, j = 0, n_{cp}\}$
Local level	BC1	$S_{C,i} = \{v_{i,0}\}, i = 1,...,n_v$
	BC2	$S_{C,i} = \{v_{i,0}\}, i = 1,...,n_v$
	BC3	$S_{C,i} = \phi, i = 1,...,n_v$, ϕ is an empty set

Since the initial and final positions of the *VL* (or the formation) are know the first and last control points are known and not optimized (Xu, Remeikas, and Pham 2011)

Set $S_{O,VL}$ (Table 1) includes the parameters that are optimized in the cooperative level optimization of *VL*. The cooperative level planning is formulated as the following NLP (**P1**):

$$\min_{x_{VL,p,j} \in S_{O,VL}} J = \sum_{i=1}^{n_v} J_i, \quad j = 1,...,n_p \tag{27}$$

such that

$$\begin{cases} V_{VL,k}^2 \le V_{i,\min}^2 + \underline{\lambda}_k \\ V_{VL,k}^2 < V_{i,\max}^2 + \overline{\lambda}_k \\ \boldsymbol{g}_{VL}(\boldsymbol{x}_{VL}, \boldsymbol{u}_{VL}, \boldsymbol{X}, t) \le 0 \end{cases}, \quad \begin{matrix} i = 1,...,n_v \\ k = 0,...,N \end{matrix} \tag{28}$$

, in which J_i is the performance index optimized in the local level NLP.

Only the total energy is used as the performance index in this paper. However as mentioned earlier, other performance index can be used as well.

Based on Assumptions 2 and 3, the state and control variables of each follower at the discretized nodes can be calculated via the differential inclusion (Kumar and Seywald 1996). The local level trajectory planning for follower i is formulated as the following NLP (**P2**):

$$\min_{v_i \in S_{O,i}} J_i = \sum_{k=0}^{N} \boldsymbol{u}_{i,k}^T \boldsymbol{u}_{i,k} w_k \tag{29}$$

such that

$$\boldsymbol{g}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, t) \le 0, \; i = 1,...,n_v \tag{30}$$

in which w_k is the weight of the node k in the discretization (Hesthaven, Gottlieb, and Gottlieb 2007, Fahroo and Ross 2001).

Feasible and Optimal Cooperative Trajectory Generation

The feasible solution and the optimal solution can be found using Algorithm 2 shown below.

Table 3 Algorithm 2: Feasible and optimal cooperative trajectory planning algorithm

Step 1:	Use Algorithm 1 to find a collision free and smooth *VL* path.
Step 2:	Start to solve **P1** (Step 2 to Step 13).
Step 3:	Use the parameters in Set $S_{O,VL}$ found in Step 1 or generate new parameters in $S_{O,VL}$ as the initial guess for **P1**.
Step 4	If all the early termination conditions (Eqs. 5, 39, and 40) are satisfied, continue to Step 5. Otherwise, go back to Step 3.
Step 5:	The *VL* will pass t_0, t_f, Δ_i, "yes/no", and $\boldsymbol{x}_{VL,p}$ to the followers. Start to solve **P2** in a decentralized manner by the followers. (Step 5 to Step 11)
Step 6:	Generate parameters in Set $S_{O,i}$ as the initial guess for **P2**.
Step 7:	Calculate the parameters in $S_{O,i}$ using the appropriate equations derived in Section 4.B.
Step 8:	Compute the state and control variables using the *VL* path, SCPs, and desired formation information.
Step 9:	Evaluate the performance index J_i and the constraints in **P2**.
Step 10:	If the maximum number of iterations has not been reached and the convergence criterion is not yet satisfied, go back to Step 6. Otherwise, continue to Step 11.
Step 11:	The optimization of **P2** is terminated. If the optimization of P2 is successful and the goal is to find a feasible solution, **the algorithm stops and the result achieved is a feasible solution.** If the optimization of **P2** is not successful or the goal is to find the optimal solution, continue to Step 12.
Step 12:	Followers transmit J_i and $\boldsymbol{x}_{i,p}(t), t \in [t_0, t_s]$ information to the *VL*.
Step 13:	If the performance index $J = \sum_{i=1}^{n_v} J_i$ can be improved, or certain constraints are not satisfied. Go back to Step 3. Otherwise, **the algorithm stops and the optimal solution is obtained.**

It is worth noting that the convergence of the NLP considering different constraints is not guaranteed. Since the initial guess is simply an obstacle free path from a grid-based search in Algorithm 1, the solution may only be locally optimal around this initial condition. If an optimal solution that meets all the inequality and equality constraints cannot be found, a different obstacle free path may be used.

Problem Dimension, Optimality, and Communication Analysis

Problem Dimension

In Algorithm 2, the wavefront problem and three NLPs (**P1**, **P2**, and **P3**) are solved. Only one backward propagation and one forward propagation are involved in the basic wavefront method applied here. Since no iterations are involved and the complexity of the wavefront algorithm is linearly related to the number of grids (Ozdal and Wong 2009), the computational complexity of the wavefront algorithm is much lower than that of the NLPs. Therefore only the problem dimensions of those three NLPs are considered (Xu, Remeikas, and Pham 2011).

The numbers of parameters to be optimized in **P1**, **P2**, and **P3** are on the order of $\mathrm{O}(n_p n_{cp})$, $\mathrm{O}(N)$, and $\mathrm{O}(n_p n_{cp})$, respectively. Therefore, the problem dimension of the achieved NLPs in finding the feasible solution is on the order of $\mathrm{O}\left\{2n_p n_{cp} + N\right\}$, while that of the optimal solution is on the order of $\mathrm{O}\left\{n_p n_{cp} + n_p n_{cp} N\right\}$.

An NLP may have a polynomial to exponential time complexity (Rao, Wright, and Rawlings 1998). Let us use the exponential time complexity as an example. The time complexity of the feasible solution algorithm is on the order of $\mathrm{O}\left\{2^{n_p n_{cp}} + 2^{n_p n_{cp}} + 2^N\right\}$, while the time complexity of the optimal cooperative algorithm is on the order of $\mathrm{O}\left\{2^{n_p n_{cp}} + 2^{n_p n_{cp}} \cdot 2^N\right\}$.

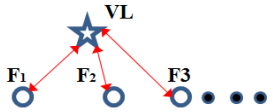
Optimality

Theorem 3: If Algorithm 2 converges, the limiting cooperative trajectory obtained ($\{\boldsymbol{v}^*\}_N^{n_{cp}}$, $\{\boldsymbol{x}_{i,p}^*\}_N^{n_{cp}}$, and $\{\boldsymbol{x}_{VL,p}^*\}^{n_{cp}}$) equals to the optimal solution ($\{\boldsymbol{v}^*\}$, $\{\boldsymbol{x}_{i,p}^*\}$, and $\{\boldsymbol{x}_{VL,p}^*\}$), when the number of control points N and the number of discretization nodes n_{cp} increase to infinity.

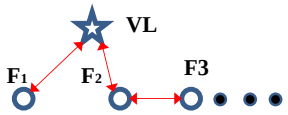
Based on the above analysis shown in (Xu, Remeikas, and Pham 2011), as the numbers of discretized nodes and the control points increase to infinity, if Algorithm 2 converges, the solution is the optimal one. It is worth noting that this solution is only locally optimal rather a global optimum.

Virtual Leader-Follower Communication

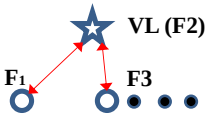
The cooperative system is comprised of one *VL* and multiple followers F_i, $i = 1,...,n_v$. The communication architecture is shown in Fig. 3, in which each follower is assumed to be able to communicate directly with the *VL* (Fig. 3a) or indirectly with the *VL* through other followers (Fig. 3b) if this follower is not within the communication range of the *VL* (Xu, Remeikas, and Pham 2011). The communication graph is assumed to be connected. In the case when the *VL* is malfunctioning, one of the followers will be selected as the *V*L and the communication architecture will be rearranged (Fig. 3c).



(a) Communication between the *VL* and followers.



(b) Communication between the *VL* and followers via other followers.



(c) Communication architecture rearranged.

Figure 3 Communication topology

The communication complexity (Dunbar 2007) depends on the total information transmitted between the *VL* and the followers.

The information flowing from the *VL* to n_v followers includes: (i) "Yes" or "No" (a total of n_v variables for n_v followers) about whether the cooperative level optimization is accomplished; (ii) desired formation geometry (a total of $n_v n_p$ variables), path of the *VL* (a total of $n_v n_p(1+n_{cp})$ variables), and initial and final time ($2n_v$ variables).

The information the *VL* received from n_v followers. (i) The performance index J_i (a total of n_v variables). It is worth noting that a predefined number can be used if the local level optimization is not successful. (ii) The trajectory information between $[0, t_s]$. If there are N_s nodes within this time frame, the number of variables transmitted is $n_v n_p N_s$.

Therefore, the communication complexity for the *VL* is bounded by $[4+(2+n_{cp}+N_s)n_p]n_v$ variables, and the communication complexity bound for each follower is $4+(2+n_{cp}+N_s)n_p$. For example, if a wireless communication device has a bit rate of 9.6Mb/s, theoretically the information update between the *VL* and the followers can be up to 5 KHz, if $n_v = 5$, $n_{cp} = 6$, $n_p = 2$, $N_s = 3$, and the variable is double precision. Thus the communication complexity is low in the proposed algorithms.

As mentioned in (Belta and Kumar 2004), the proposed virtual leader-follower architecture is a centralized approach. However, since each vehicle only needs to know its own states and the *VL*'s information, the communication complexity is not high. Therefore the algorithm scales well as the number of robots increases.

Although in this paper network-induced constraints such as delays or packet losses are not considered, they are worth mentioning. These issues could potentially be addressed in future work by incorporating the methods described in (Wu & Shi 2011).

As shown in Corollary 1, followers are only required to communicate a part of their position information with the *VL* for the purpose of checking the inter-vehicle collision during the transient stage. Once the formation is achieved, the inter-vehicle collision is guaranteed to be avoided in the current planning horizon and information transmission is not required.

The information among the *VL* and the followers are updated at the following instants: (i) the beginning of each trajectory planning horizon t_0, (ii) any time when a follower has finished its local level optimization, and (iii) any time when new obstacles or unexpected events pop up.

CHAPTER FOUR: SIMULATION EXAMPLES

Minimal Energy Trajectory Generation for a Group of Two-Wheel Drive Robots

Simulation Settings

To test the effectiveness and robustness of the algorithms, a minimal energy trajectory planning scenario for a group of cooperative robots is simulated (Xu, Remeikas, and Pham 2011). The NLPs in the algorithms are solved using the "fmincon" function in Matlab (Version 7.10.0.499-R2010a). The constraint and function evaluation tolerances for the VL and followers are set to 10-3 and 10-3, respectively. A desktop computer with a 3.10GHz CPU and 4 GB of RAM is used for all of the computations. The motion of the two-driving wheel robots in the formation is driven by the following nonlinear dynamics (Laumond, Sekhavat, and Lamiraux 1998)

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} \cos\theta_i \\ \sin\theta_i \\ 0 \end{bmatrix} V_i + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega_i, \ i = 1, ..., n_v \tag{31}$$

where the midpoint of the two wheels defines the position state by $\boldsymbol{x}_{i,p} = [x_i, y_i]^T$. Here the speed V_i and the angular velocity ω_i are the control variables and constrained by $|\omega_i| \le \omega_{i,\max}$ (e.g. $\omega_{i,\max} = 180^\circ / s$) and $0 \le V_i \le V_{i,\max}$ (e.g. $V_{i,\max} = 3m/s$). The objective of the simulation is to move a formation of robots from the bottom left corner to the top right corner of the defined area using the least possible amount of total energy. The cost function associated wollower i in its discretized form is

$$J_i = 0.5(t_f - t_0)\sum_{k=0}^{N}(m_i V_{i,k}^2 + I_i \omega_k^2) w_k \tag{32}$$

The mass m_i and the inertia I_i are assumed to be equal to 1. The cost function associated with the overall formation is given by $J = \sum_{i=1}^{n_v} J_i$. The control variables and state rate are calculated by $V_i = \sqrt{\dot{x}_i^2 + \dot{y}_i^2}$, $\theta_i = \tan^{-1}(\dot{x}_i / \dot{y}_i)$, $\omega_i = (\ddot{y}_i \dot{x}_i - \ddot{x}_i \dot{y}_i) / (\dot{x}_i^2 + \dot{y}_i^2)$, respectively.

A series of Monte Carlo simulations are conducted and analyzed to test the robustness of the algorithm numerically. In these simulations, the following settings are varied: the number, position, and size of the obstacles; the initial and final positions of the *VL* and the followers; and the formation of the followers. The number of obstacles varies from 10 to 30 with the radius in a range from 5m to 15m. The locations of the obstacles are varied randomly throughout the inner area of 200m x 200m. The formation is in a set shape but the position bias between the followers and the leader in the formation location will be varied from 0m to 10m. The initial and final positions of the *VL* leader and followers are chosen from a 15m x 15m area in the bottom left and top right corners of the given area, respectively. The rest of the settings in the Monte Carlo runs are static: the size of the area is 250m x 250m, the number of followers is 3 or 5, the number of discretization nodes is 10, the initial SPC guess for each follower is set to be 2, the degree and number of control points used for the b-Spline are 5 and 6, respectively, and the bounds on the SPC, speed, and turn rate are set to be [1,10], 3m/s, and $180^\circ / s$, respectively.

To show the scalability of the proposed algorithms, the cases with three follower and five follower robots are tested in the Monte Carlo simulation and the computational time is compared.

Three-Follower Case

Three followers are required to form and maintain their desired formation with respect to the *VL*, while meeting all of the constraints. Two types of solutions are shown: the feasible and the optimal. In the feasible solution, the algorithm stops after the first trajectory, which satisfies all of the constraints is found. In the optimal solution, the algorithm will continue to iterate until the minimal performance index is obtained.

The feasible and optimal solutions are compared by their CPU time and performance index (PI). Table 5 shows the average results from a 1000-run Monte Carlo simulation. The feasible method produces a solution in a much faster time, less than half a second, while the optimal solution takes around 33 seconds. The optimal solution provides an average improvement to the PI of about 13% when compared to the feasible solution. It is worth noting that the CPU time can be reduced further without saving and plotting all the data during the Monte Carlo runs.

Table 4 Simulation results for three followers from 1000 Monte Carlo runs

Solution	Avg. CPU Time (s)	Avg. % CPU Time Ratio	Avg. % PI Improvement
Feasible	0.45	1.35	13.21
Optimal	33.23		

The trajectories generated from a randomly selected result are shown in Fig. 4. Each follower is represented in a different line style and the obstacles are the circular objects. The formation takes shape after a short transient stage, while the time after it is reached is the formation steady stage. During both of these stages, inter vehicle collision is avoided.

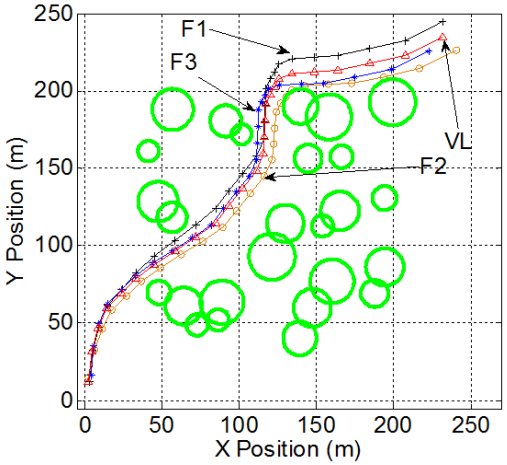


Figure 4 Three-follower case forming and maintaining a formation

In Figs. 5 and 6 the control variables of each follower are shown as a function of time. Both of these varabiles stay within the required constraints mentioned previously.

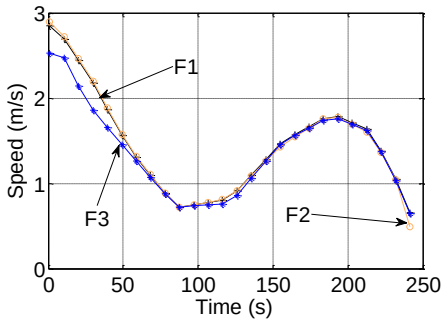


Figure 5 Followers' speed profiles for the three-follower case.

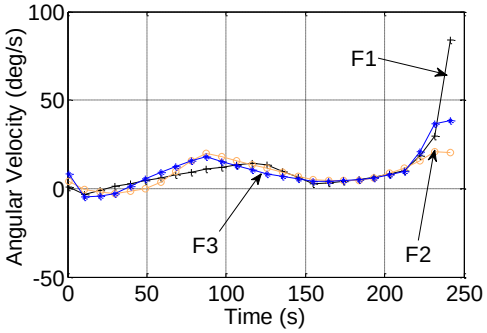


Figure 6 Followers' angular velocity profiles for the three-follower case.

Each of the followers may reach their desired location in the formation at a separate time and in a decentralized manner. The speed of the follower reaching its desired formation is controlled by the SPC. In Fig. 7 the SPC of each follower is shown over the course of the trajectory, staying positive and within the bounds set.

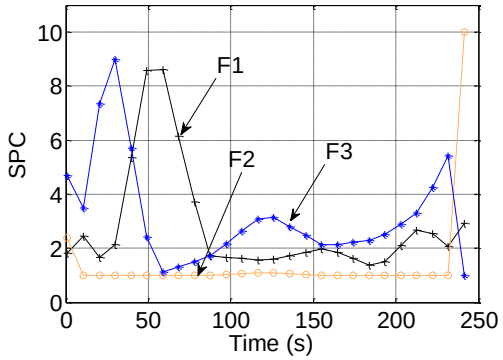


Figure 7 Followers' SCP profiles for the three-follower case.

Five-Follower Case

Another Monte Carlo simulation is conducted with five followers. The same settings are used as the three-follower case. The results shown in Table 6 are similar to the three-follower case in that when using the feasible method the solution is found in a very small amount of time. The optimal solution also took longer as expected but reduced the performance index by a noticeable amount.

Table 5 Simulation results for five followers from 1000 Monte Carlo runs

Solution	Avg. CPU Time (s)	Avg. % CPU Time Ratio	Avg. % PI Improvement
Feasible Optimal	0.49 36.54	1.34	15.20

Figure 8 shows a selected case from the five-follower Monte Carlo simulation. The formation is maintained with five followers throughout the trajectories. In Fig. 9 inter-vehicle collision avoidance is shown in the zoomed-in figure. After this section the formation is quickly reached and maintained.

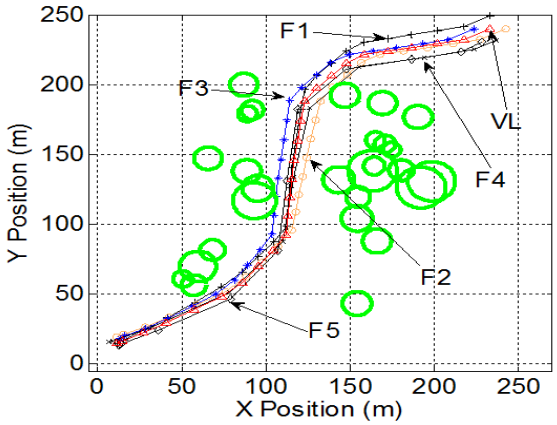


Figure 8 Five-follower case forming and maintaining a formation.

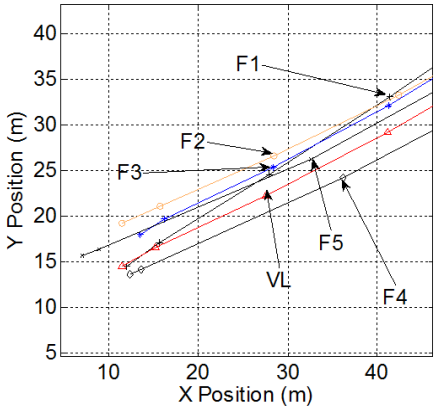


Figure 9 Zoomed-in formation transient stage in the five-follower case.

In Figs. 10-12, the control variables and the SCP are shown. Similar arguments can be obtained as those of the three-follower case.

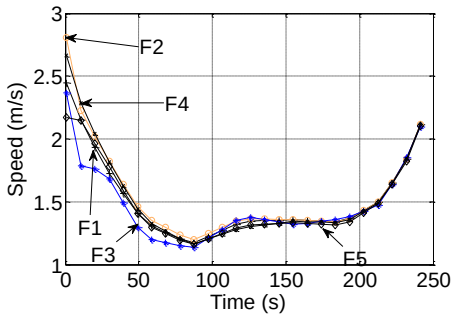


Figure 10 Followers' speed profiles for the five-follower case.

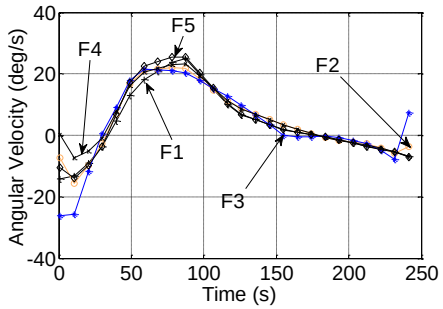


Figure 11 Followers' angular velocity profiles for the five-follower case.

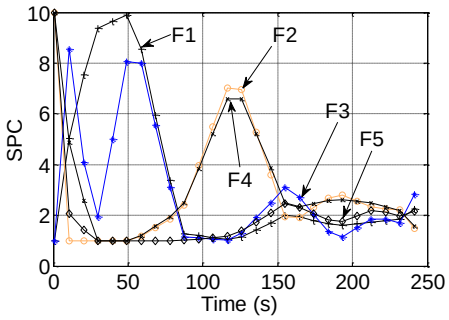


Figure 12 Followers' SCP profiles for the three-follower case.

To show the scalability of the algorithm, the Monte Carlo simulations from the three- and five-follower cases are compared in Table 6. In almost all of the cases the change in CPU time is less than 10%. This shows that the number of followers has very little effect on the overall computation cost of the algorithm for both the feasible and optimal solutions. In addition, both the three-follower and five-follower have very consistent robustness with a consistent success

rate above 90%. It is worth noting that the unsuccessful runs are due to the improper settings of the obstacles, or no feasible path/trajectory for the formation to go through.

Table 6 Scalability and robustness of the algorithms

	Avg. CPU Time (s)		Time Diff. %	
# of followers	Feasible	Optimal	Feasible	Optimal
3	0.45	33.23	8.59	9.98
5	0.49	36.54		
	Success Rate			
	Feasible		Optimal	
3	90.1%		90.7%	
5	90.3%		90.4%	

To verify the accuracy of the discretization scheme used in the open loop control design, the open loop control commands generated are interpolated and used to forward propagate the robot dynamics Eq. (31). The mismatches of the boundary conditions and the trajectories between the planned one and the forward propagated one are small.

Farming Task Assignment and Trajectory Planning

Due to the increased interest in automated farming, many studies have been conducted on increasing the efficiency and precision of different farming task. In (Eaton et al. 2008) a framework is present for an autonomous farming that is presented as a system of systems. In this framework everything from path planning, low level tracking, to precision seeding are covered (Remeikas et al. 2013). In addition to this type of research, a lot of papers focus on the small aspects of handling different portions of automated farming. In (Kazmi et al. 2011), a method to detect, determine task, and execute these task in disease detection is presented. The system consists of various aerial and ground vehicles to accomplish the different requirements of the

problem. With the current trend in almost all systems moving towards automation, it is logical that agriculture would be interested. As shown in (Pedersen et al. 2006) that is would be beneficial for the agricultural industry to move towards automation due to the ability to reduce the cost of labor, replace trivial task with autonomous systems, and increase the accuracy of various activities.

Background

As mentioned before, a realistic example of a simple citrus harvest example is studied. An iterative hierarchical cooperative planning framework is developed for a group of agricultural robots. The framework is broken down into the two levels, the cooperative level where the formation configuration is selected and the leader's trajectory is generated, and the lower level where the follower's trajectories are generated.

Farming Vehicle Model

There are n_R robots in the cooperative farming system and the motion of each agriculture robot i is described by its own nonlinear dynamics, for example the following 2-D nonlinear model.

$$\begin{cases} \dot{x}_i = V_i \cos\theta_i \\ \dot{y}_i = V_i \sin\theta_i \\ \dot{\theta}_i = \left(V_i / L_i\right)\tan\delta_i \\ \dot{\delta}_i = u_i \end{cases}, \quad i = 1,...,n_R \tag{33}$$

The state vector of robot i includes the position vector of the geometric center $\boldsymbol{x}_i = [x_i, y_i]^T \in \Re^{2\times 1}$, the heading angle θ_i, and the angle of the front steered wheel δ_i. The

length from the front wheel to the rear wheel is represented by L_i. In this model, the steering is handled by a single front wheel while the rear wheels provide the drive. The control variables are the speed V_i and the turning angular rate of the front wheel steering angle $u_i = \dot{\delta}_i$. This model does not explicitly consider the effects of slip from the robot, instead it can be considered as an inequality constraint if necessary.

Simulation Results

The objective of the simulated agricultural robots is to harvest two parallel curves of citrus with a minimum time in the cooperative level and minimum energy consumption in the individual level. The software is programmed in MATLAB (Version 7.8.0-R2009a) on a desktop computer with a 2.33 GHz CPU and 2.95 GB of RAM. In the smoothing operation, the constraint and function tolerances are set as 10^{-12} and 10^{-1}, respectively, while those of the local level robot trajectory generation are set to both be 10^{-4}.

To test the capability of using the algorithm for heterogeneous models (Eq. 40), three agricultural robots will have different sizes and dynamic characteristics as shown in Table 1.

Table 7 Parameters for three agricultural robots

Agricultural Robot	1	2	3
L_i	4 m	4.5m	5m
m_i	19,530kg	21,700kg	23,870kg
I_i	55,335$kg\ m^2$	61,843 $kg\ m^2$	67,631 $kg\ m^2$
$V_{i,\max}$	0.22 m/s	0.25 m/s	0.19 m/s
$\dot{\theta}_{i,\max}$	$3°/s$	$5°/s$	$5°/s$
$\dot{\delta}_{i,\max}$	$0.3°/s^2$	$0.5°/s^2$	$1.0°/s^2$

The farming area in the simulation has a dimension of 125m by 60m. The areas occupied by citrus to be harvested by the robots are represented by circles with a radius of 1.5m. A total of 22 plants are placed in the farming area, in two parallel curves. The rows are separated by an average distance of 7m, while each plant has 4m between them in the row, and an overview of the farming setup is shown in Fig 14. In addition, there will be several obstacles (i.e. fallen trees or stones) directly in the path of the robot.

Three formation configurations are assumed to be proper in harvesting the rows of plants as shown in Fig. 15. As mentioned before, an example formation can be a right triangle shape where one tractor follows behind to collect fallen fruits.

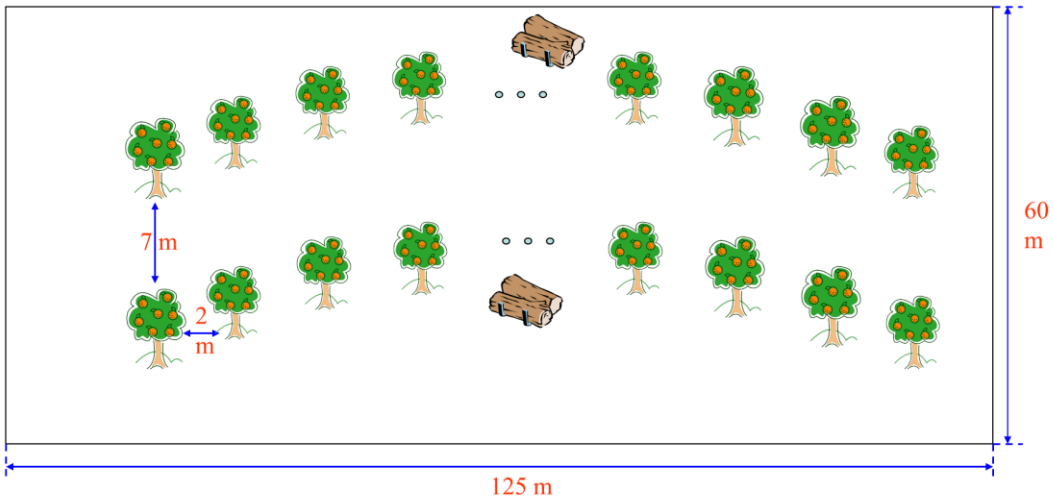


Figure 13 Citrus farm layout used in the simulation

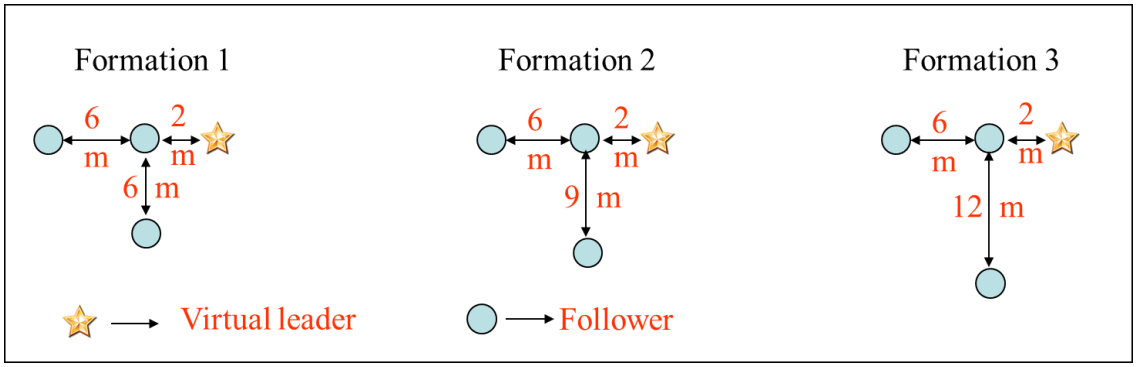


Figure 14 Three available formation configuration options in the simulation

The performance index shown in Eq. (7) for the Cooperative Level Algorithm is utilized for the cooperative system to help with ranking the formation configurations and position locations. In this simulation the weights W_b and W_f are both set to be one to provide an equally weighted case on the formation transient times and the cooperative minimal distance. In the robot cooperative trajectory optimization, the performance index shown in Eq. (13) is used.

In the cooperative level optimization, the wavefront path planning algorithm is used to

provide an obstacle free corridor. The number of grids used to represent the farming area is 100 x 100. To provide the smooth trajectory a B-spline representation with the number of control points and degree set to be 2 and 3, respectively, is used. In the individual level optimization, the robots trajectories are generated using the Individual Level Algorithm. This trajectory planning time horizon is discretized into 9 nodes with the trivial initial guess for the SCP at each node being 5. The safety buffer to prevent inter-vehicle collision, ε_s , is set to be 1m.

The simulated citrus harvesting task is broken down into seven phases. In the first phase, the robots are traveling along the first row of trees. In the second phase, the formation is changed to avoid an additional obstacle. In the third phase, the robots continue moving along the first row of trees. In the fourth phase, the robots move from the first row to the second row. In the fifth phase, the robots will move along the second row of trees until an obstacle is encountered. In the sixth phase, the robots will avoid the fallen plant and then, in the last phase, the three robots move along the rest of the second row of trees before completing the task. The end of each phase can be viewed as a discrete event that will trigger the re-start of the decision making and planning algorithm. From the three formation configurations and the available formation location positions, there are three options per formation configuration as shown in Fig. 15. Each phase of the task is analyzed and a summary of the results are shown.

For each phase, the virtual leader's path is generated using Cooperative Level Algorithm. In Table 8, a summary of the CPU time spent during the leader path generation is shown. It can be seen that the CPU in generating the leader's path is mostly less than 0.3 seconds except one case.

Table 8 CPU time spent in generating the leader's path (Seconds)

	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6	Phase 7
Leader's Path Generation							
Formation Configuration 1 Computed by follower 1	1.463	0.330	0.105	0.071	0.092	0.0931	0.141
Formation Configuration 2 Computed by follower 2	0.178	0.093	0.084	0.073	0.097	0.066	0.088
Formation Configuration 3 Computed by follower 3	0.151	0.093	0.035	0.066	0.095	0.095	0.091

A more in-depth look at the CPU time spent in Phase 1 by each follower robots for both parts of Cooperative Level Algorithm is shown in Table 9.

Table 9 CPU times of the Cooperative Level Algorithm in Phase 1 (seconds)

	Part 1	Part 2
Formation Configuration 1 by follower 1	0.034	1.277
Formation Configuration 2 by follower 2	0.0175	0.135
Formation Configuration 3 by follower 3	0.0179	0.116

The Formation Ranking Algorithm is used to determine the formation configuration and position location rankings. The CPU time used in this centralized step is shown in Table 10 and it is obvious that the ranking can be accomplished in a very short time.

Table 10 CPU times used in formation ranking (Seconds)

	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6	Phase 7
CPU Time	0.0130	0.0034	0.0004	0.0004	0.0005	0.0004	0.0004

Finally, the individual level optimization is used to generate the optimal trajectories for the follower robots. The CPU time spent in this algorithm is listed in Table 11, and it can be seen that the optimization can be done rapidly.

Table 11 CPU time used in the Local Level Algorithm

	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6	Phase 7
Follower 1	0.334	0.028	0.021	0.016	0.018	0.179	0.020
Follower 2	0.251	0.016	0.017	0.019	0.020	0.031	0.017
Follower 3	0.092	0.016	0.18	0.015	0.021	0.020	0.019

Considering that the Cooperative Level Algorithm and Local Level Algorithm are decentralized, the total CPU time used for all seven phases is only about 2.6 seconds.

The trajectories generated for all seven phases are shown in Figs. 16 and 17. Each of the followers is represented as a different line style while the plants are the circular objects. The follower robots quickly reach the desired position locations in the formation and proceed to maintain the formation.

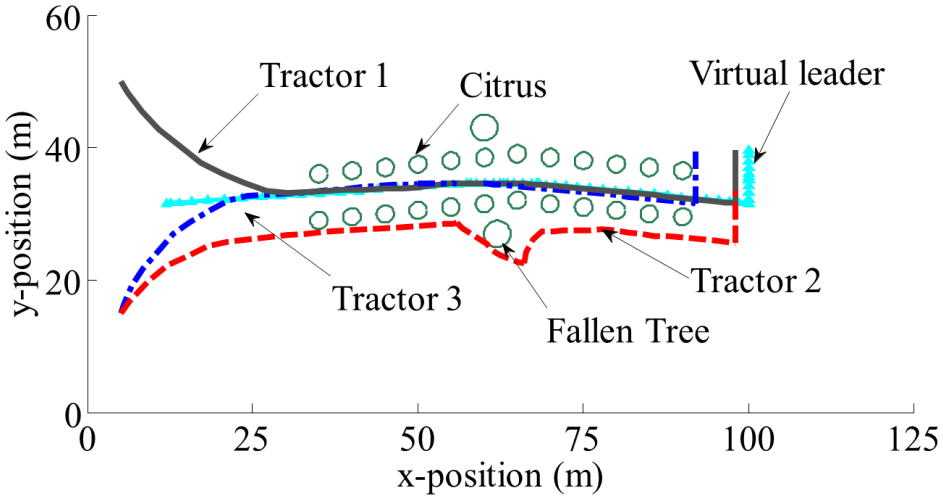


Figure 15 Generated trajectories for the followers and virtual leader in Phases 1 through 4

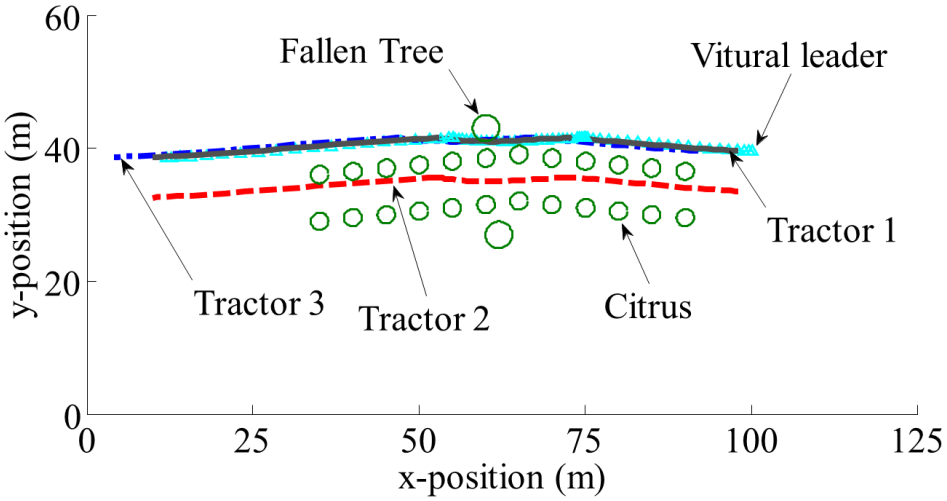


Figure 16 Generated trajectories for the followers and virtual leader in Phases 5 through 7

In Figs. 18 and 19, the control variables, the speed, and the steering angle are shown. These stay within their constraints mentioned in the simulation settings.

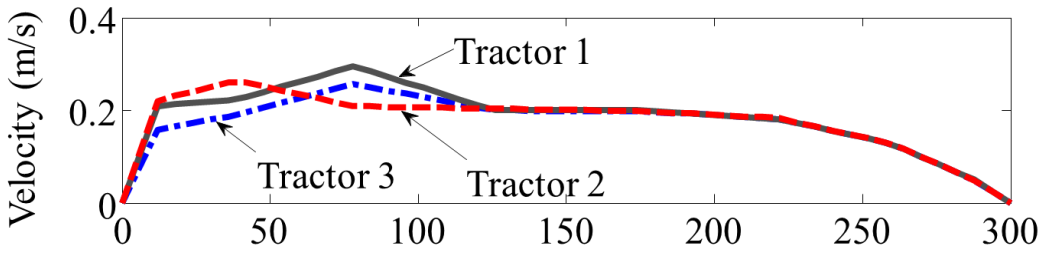


Figure 17 Followers' speed profiles during Phase 1.

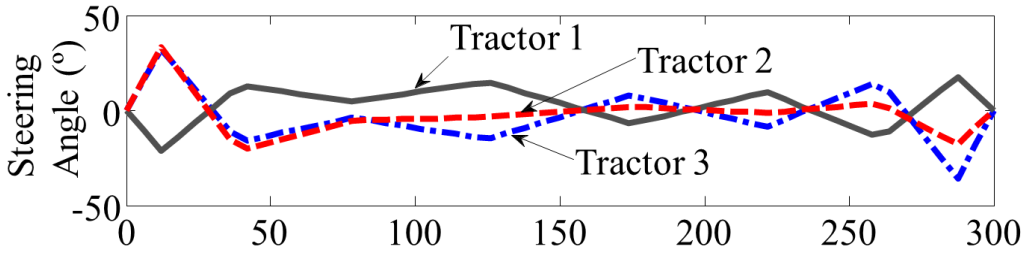


Figure 18 Followers' steering angle profiles during Phase 1.

The heading angle and speed control parameter plots are showing in Fig. 20 and 21. Once the robots have reached their formation positions the plots converge as they are all moving in a common direction.

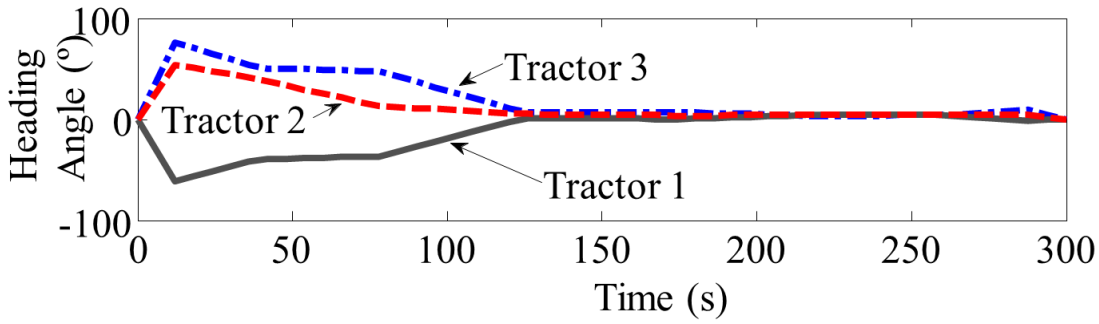


Figure 19 Followers' heading angle profiles during Phase 1.

Since the leader's path is parallel with the x direction, the followers' heading angles converge to zero during the majority of their trajectory. During the transient stage the followers turn to reach their desired position locations.

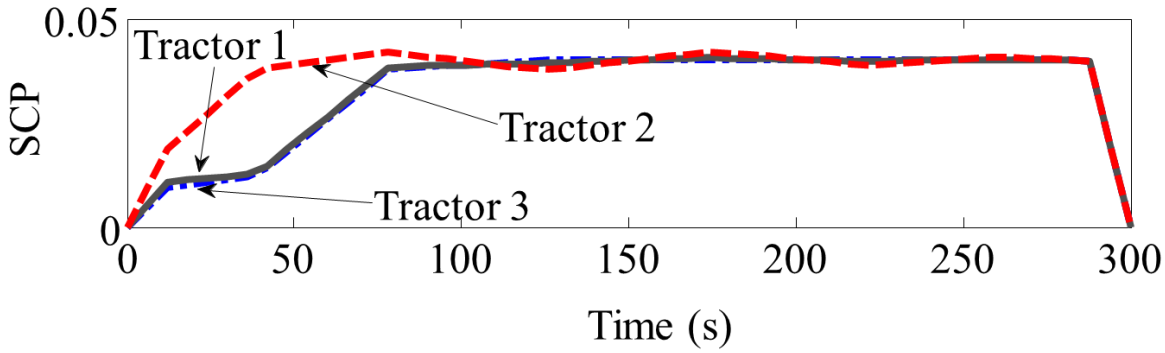


Figure 20 Followers' SCP profiles during Phase 1

Similar results can be drawn from phases 2 through 7 but are omitted for brevity.

CHAPTER FIVE: SUMMARY AND FUTURE WORK

Summary

There are many difficulties associated with generating trajectories for a group of cooperative vehicles. A method to rapidly generate these considering complex dynamics and severe constraints is presented and analyzed. The major benefits of this approach are that the formation can be globally asymptotically maintained, there is very little information required to generate the followers' trajectories, the problem dimension is reduced via the local pursuit strategy, and it has the ability to handle nonlinear heterogeneous dynamics with severe constraints. Due to the formulation and methods used, such as early termination conditions, the computational cost of the trajectory generation in the framework is low.

Two simulation examples are presented to showcase the ability of the framework to produce these results with all of the expected benefits. Monte-Carlo simulations on a group of cooperative ground vehicles are shown to verify that the system is capable of quickly generating trajectories in very complex environments with a varying number of vehicles. A realistic citrus harvest path planning example is investigated using the studied methods that shows the system can cooperatively decide on the desired formation configurations and also rapidly produce the individual trajectories.

Future Work

Currently, research is being conducted on applying this framework to plant disease detection and monitoring system. With a quadrotor acting as the virtual leader in the system, it can detect

and process the information to be sent to the lower level vehicles such as geography and obstacle information. The followers in the system are simple ground robots able to detect and collect the disease infected parts of the plants to be brought back to a lab for analysis. Another example of a system that this framework could be applied to is formation flight, for applications like surveillance where the cooperative goal is to survey a wide area. Different formations could be used that would allow for maximization of coverage area under certain restrictions and constraints if the airspace is limited or crowded. Since in the lower level, the vehicles are subjected to a performance index, fuel consumption could be minimized to reduce the overall cost of the operation.

There are still areas that could be improved upon in the current method. The assumptions made about the dynamics can be investigated as these are still limiting the available systems that can be analyzed. While the method presented in this paper considers heterogeneous nonlinear dynamics, it has the restriction that the state and control variables must be able to be calculated directly using dynamic inclusion or inversion.

LIST OF REFERENCES

Bajodah, A.H., (2009), 'Generalised Dynamic Inversion Spacecraft Control Design Methodologies,' *IET Control Theory & Applications*, 3, pp. 239-251.

Belta, C., and Kumar, R. V., (2004), 'Abstraction and Control for Groups of Robots,' *IEEE Transactoins on Robotics*, 20, pp. 865-875.

Bernstein, D. S., (2005), Matrix Mathematics – Theory, Facts, and Formulas with Application to Linear Systems Theory, Princeton University Press., pp. 271.

Blackmore, B. S., Fountas, S., Vougioukas, S., Tang, L., Sørensen, C. G., and Jørgensen, R., "A method to define agricultural robot behaviors," Mechatronics & Robotics Conference (MECHROB) 2004, pp.1197-1200.

Cormen, T., Leiserson, C., Rivest, R., and Stein, C., Introduction to Algorithms, Cambridge, Massachusetts, The MIT Press, 2009.

Cortes, J., Martinez, S., and Bullo, F., (2006), 'Robust Rendezvous for Mobile Autonomous Agents via Proximity Graphs in Arbitrary Dimensions,' *IEEE Transactions on Automatic Control*, 519, pp. 1289-1298.

Desai, J., Ostrowski, J., and Kumar, V., (2001), 'Modeling and Control of Formations of Nonholonomic Mobile Robots,' *IEEE Transactions on Robotics and Automation*, 17, pp. 905-908.

Dong, W. J., and Farrell, J. A., (2008), 'Cooperative Control of Multiple Nonholonomic Mobile Agents,' *IEEE Transactions on Automatic Control*, 53, pp. 1434-1448.

Dunbar, W. B., (2007), 'Distributed Receding Horizon Control of Dynamically Coupled Nonlinear Systems,' *IEEE Transaction on Automatic Control*, 52, pp. 1249-1263.

Dunbar, W. B., and Murray, R. M., (2004), 'Receding Horizon Control of Multi-Vehicle Formations,' *43rd IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, pp. 1995-2002.

Dunbar, W. B., and Murray, R. M., (2002), 'Model Predictive Control of Coordinated Multi-Vehicle Formation,' *In the 41th IEEE Conference on Decision and Control*, Las Vegas, NV, pp. 4631-4636.

Eaton, R., Katupitiya, J, Pota, H., and Siew, K. W., "Robust sliding mode control of an agricultural tractor under the influence of slip", *International Conference on Advanced Intelligent Mechatronics*, Singapore, July 14-17, 2009, pp. 1873-1878.

Fahroo, F., and Ross, I. M., (2001), 'Costate Estimation by a Legendre Pseudospectral Method,' *Journal of Guidance, Control, and Dynamics*, 24, pp. 270-275.

Ferrari, C., Pagello, E., Ota, J., and Arai, T., (1998), 'Multirobot Motion Coordination in Space and Time,' *Robotics and Autonomous Systems*, 25, 219-229.

Girard, A. R., de Sousa, J. B., and Hedrick, J. K., (2001), 'An Overview of Emerging Results in Networked, Multi-Vehicle Systems,' *In the 40th IEEE Conference on Decision and Control*, Orlando, FL, pp. 1485–1490.

Gong, Q., Ross, I. M., Kang, W., and Fahroo, F., (2008), 'Connections between the Covector Mapping Theorem and Convergence of Pseudospectral Methods for Optimal Control,' *Computational Optimization and Applications*, 41, pp. 307-335.

Goodwin, G. C., Graebe, S. F., and Salgado, M. E., (2001), Control System Design, Prentice Hall, Upper Saddle River, NJ.

Guarino Lo Bianco, C., and Piazzi, A., (2002), 'A Servo Control System Design using Dynamic Inversion,' *Control Engineering Practice*, 10, pp. 847-855.

Hamner, B., Bergerman, M., and Singh, S., "Specialty crop automation with Autonomous vehicles." *International Conference on Robotics and Automation*, St. Paul, MN, 2012, pp. 1829-1835.

Hesthaven, J., Gottlieb, S., and Gottlieb, D., (2007), Spectral Methods for Time-Dependent Problems, Cambridge, UK: Cambridge University Press.

Hristu-Varsakelis, D., and Shao, C., (2004), 'Biologically-Inspired Optimal Control: Learning from Social Insects,' *International Journal of Control*, 77, pp. 1549-1566.

Jin, L., and Tang, L., "Optimal coverage path planning for arable farming on 2D surfaces," *Transactions of the American Society of Agricultural and Biological Engineers*, Vol. 53, No.1, 2010, pp. 283-295.

Karkee, M., "Modeling, identification, and analysis of tractor and single axle toward implement system", Iowa State University, Dissertation, 2009.

Kim, Y. S. and Mesbahi, M., (2006), 'On Maximizing the Second Smallest Eigenvalue of a State Dependent Graph Laplacian,' *IEEE Transactions on Automatic Control*, 51, pp. 116-120.

Kumar, R. R. and Seywald, H., (1996), 'Should Controls be Eliminated While Solving Optimal Control Problems via Direct Methods?' *Journal of Guidance, Control and Dynamics*, 19, pp. 418-423.

Laumond, J. P., Sekhavat, S., and Lamiraux, F., (1998), 'Guidelines in Nonholonomic Motion Planning for Mobile Robots,' *Lecture Notes in Control and Information Sciences*, LNCIS 229, New York: Springer-Verlag, pp. 1-44.

Li, M., Imou, K., Wakabayashi, K., and Yokoyama, S., "Review of research on agricultural vehicle autonomous guidance," *International Journal of Agricultural and Biological Engineering*, Vol. 2, No. 3, September 2009, pp. 1-26.

Linker, R., and Blass T., "Path-planning algorithm for vehicles operating in orchards," *Biosystems Engineering*, Vol. 101, 2008, pp. 152-160.

Liu, L., Crowe, T. G.., and Roberge, M., "Sensor-based scouting algorithms for automatic sampling tasks in rough and large unstructured agricultural fields," *Transactions of the ASABE*, Vol. 52, No. 1, 2009, pp. 285-294.

Lou, J., Cooper, J., Cao, C., and Pham, P., (2012), 'Cooperative Adaptive Control of a Two-Agent System,' *In the 2012 American Control Conference*, Montreal, Canada, pp. 2413-2418.

Marshall, J. A., Broucke, M. E., and Francis, B. A., (2004), 'Formations of Vehicles in Cyclic Pursuit," *IEEE Transactions on Automatic Control*, 49, pp. 1963-1974.

Mesbahi, M. and Egerstedt, M., (2010), Graph Theoretic Methods in Multiagent Networks, *Princeton University Press.*

Moorehead, S. J., Wellington, C. K., Gilmore, B. J., and Vallespi, C., "Automating orchards: a system of autonomous tractors for orchard maintenance," *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)* Workshop on Agricultural Robotics, 2012.

Muraro, R. P. 2009. Summary of 2008-2009 citrus budget for the southwest florida production region, University of Florida, IFAS, CREC, Lake Alfred, FL.

Murray, R. M., (2007), 'Recent Research in Cooperative Control of Multivehicle Systems,' *ASME Journal of Dynamic Systems, Measurement, and Control*, 129, pp. 571-583.

N. E. Leonard and E. Fiorelli. Virtual leaders, artificial potentials and coordinated control of groups. *In Proc. IEEE Control and Decision Conference*, pages 2968–2973, 2001.

NSF CPS Program Synopsis, http://www.nsf.gov/pubs/2010/nsf10515 /nsf10515.htm, last access on July 2012.

O'Connor, M., Bell, T., Elkaim, G., and Parkinson, B., "Automatic steering of farm vehicles using GPS," *3rd International Conference on Precision Agriculture*, Minneapolis, Minnesota, June 1996.

Olfati-Saber, R., and Murray, R. M., (2004), 'Consensus Problems in Networks of Agents with Switching Topology and Time-Delays,' *IEEE Transaction on Automatic Control*, 49, pp. 1520-1533.

Ozdal, M. M., and Wong, M. D. F., (2009), 'Global Routing Formulation and Maze Routing,' *Handbook of algorithms for physical design automation*, edited by Alpert, C. J., Mehta, D. P., and Sapatnekar, S. S., Taylor & Francis Group, Boca Raton, FL.

Parker, L. E., Birch, B., and Reardon, C., (2003), 'Indoor Target Intercept Using an Acoustic Sensor Network and Dual Wavefront Path Planning,' *In the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, pp. 278-283.

Piegl, L., and Tiller, W., (1997), The NURBS Book: Second Edition, *Springer-Verlag*, New York.

Qu, Z., (2009), Cooperative Control of Dynamical Systems: Applications to Autonomous Vehicles, Springer.

Qu, Z., Wang, J., and Hull, R. A., (2008), 'Cooperative Control of Dynamical Systems with Application to Autonomous Vehicles,' *IEEE Transactions on Automatic Control*, 53, pp. 894-911.

Rao, C. V., Wright, S. J., and Rawlings, J. B., (1998), 'Application of Interior-Point Methods to Model Predictive Control,' *Journal of Optimization Theory and Applications*, 99, pp. 723-757.

Ren, W., and Beard, R. W., (2008), 'Distributed Consensus in Multi-vehicle Cooperative Control – Theory and Application,' Springer-Verlag, London.

Remeikas C., Li, N., Xu, Y., Jayasuriya, S., and Ehsani, R., "Task Assignment and Trajectory Planning Algorithm for a Class of Cooperative Agricultural Robots," Submitted to the *ASME Journal of Dynamic Systems, Measurement, and Control*, August 2013.

Reynolds, C. W., (1987), 'Flocks, Herds, and Schools, a Distributed Behavioral Model,' *Computer Graphics*, 21, pp. 26-34.

Saber, R., and Murray, R., (2003), 'Consensus Protocols for Networks of Dynamic Agents,' *Proceedings of the American Control Conference*, 2, pp. 951-956.

Seyboth, G. S., Schmidt, G. S., and Allgower, F., (2012), 'Cooperative Control of Linear Parameter-Varying Systems,' *2012 American Control Conference*, Montreal, Canada, pp. 2407-2412.

Shiller, Z., and Gwo, Y. R., "Dynamic motion planning of autonomous vehicles," *IEEE Transaction on Robotics and Automation*, Vol. 7, Issue 2, 1991, pp. 241-249.

Srinivasan, M. V., and Davey, M., (1995), 'Strategies for Active Camouflage Motion,' *Proceedings of the Royal Society of London Biological Sciences*, 259, pp. 19-25.

Su, H., Chen, G., Wang, X., and Lin, Z., (2010), 'Adaptive Second-Order Consensus of Networked Mobile Agents with Nonlinear Dynamics,' *Automatica*, 46, pp. 368–375.

Wang, J., and Xin, M., (2010), 'Multi-agent Consensus Algorithm with Obstacle Avoidance via Optimal Control Approach,' *International Journal of Control*, 83, pp. 2606-2621.

Wu, J., and Shi, Y., (2011), 'Consensus in Multi-agent Systems with Random Delays Governed by a Markov Chain,' *Systems & Control Letters*, 60, pp. 863-870.

Xu, Y., Remeikas, C., and Pham, K., "Local Pursuit Strategy Inspired Cooperative Trajectory Planning Algorithm for a Class of Nonlinear Constrained Dynamical Systems," Accepted to the *International Journal of Control*.

Xu, Y., Xin, M., Wang, J., and Jayasuriya, S., (2012), 'Hierarchical Control of Cooperative Nonlinear Dynamical Systems,' *International Journal of Control*, 85, pp. 1093-1111.

Xu, Y., and Basset, G., (2012), 'Sequential Virtual Motion Camouflage Method for Nonlinear Constrained Optimal Trajectory Control,' *Automatica*, 48, pp. 1273-1285.

Xue, D., Yao, J., Chen, G., and Yu, Y. L., (2010), 'Formation Control of Networked Multi-Agent Systems,' *IET Control Theory and Applications*, 4, pp. 2168-2176.