Formation Control of Unmanned Ground Vehicles with Particle Swarm Optimization

Siyao Gu
Department of Electrical Engineering
Southern Illinois University

Marcos Bird, Scott Timme, Eric Ortega, Yufang Jin
Department of Electrical Engineering
University of Texas at San Antonio

Abstract

This paper investigates a formation control for a group of unmanned ground vehicles in a dynamic environment. The group consisting of one leader and three followers was arranged into a V-shaped formation. The dynamical environment includes multiple moving obstacles. Trajectory planning of the leader has been carried out for a collision free path. Formation of the group is maintained by controlling the followers to track the position of the leader. Particle swarm optimization has been integrated into the formation control to minimize the turning angle of followers. The proposed formation control scheme and trajectory planning has been verified with MATLAB simulations and application to a real platform.

Introduction

Unmanned Ground Vehicle (UGV) groups serve as powerful alternatives to personnel crews while working in a hazardous or unknown environment. Due to the complexity of the tasks assigned to the UGV group, cooperation of the group demands efficient communication and accurate formation[1]. Formation control of a system containing multiple vehicles requires making decision on more than one entity. A variety of theoretical frameworks have been proposed to formulate such problems, and various methods have been examined to solve this problem[2,3,4].

In previous research, a lead-follower control scheme had been applied to a group of UGVs consisting of one leader and three followers. The control scheme guided the UGV group with collision free path in a dynamic environment with two moving obstacles[5,6,7]. The trajectory planning was carried out by considering every vehicle in the system as one large entity and such lack of flexibility and autonomy of each vehicle. In this algorithm, the team's activities were planned based on the surrounding sensed by the leader and control of the system was almost always nonlinear. In addition, taking into account that turning of a vehicle is achieved by a rotational velocity difference between its driving motors, efficiency can be increased by minimizing the turn angle. To address these issues, particle swarm optimizing (PSO) controller has been designed and integrated into the UGV trajectory planning for the followers. PSO have been widely reported in scientific literature with applications to optimizing different variables, for example, weights of neural networks, and Markov decision processes.[8-11] This paper proposed a formation control scheme to keep V-shape of a multi-vehicle system and extended the application of PSO to minimize the turn angle of followers in a real platform. The control scheme has been verified with MATLAB simulations.

Formation Control with PSO

In our previous study of a multi-vehicle system, the leader calculated the collision-free trajectory of the whole group in a dynamic environment.[5,6,7] The followers' orientation can be determined based on the trajectory given by the leader. For any point on the trajectory, the orientation is simply the unit vector from the current position to the next checkpoint as shown in Figure 1.

$$(\hat{x},\hat{y}) = \frac{(\vec{x},\vec{y})}{\|(x,y)\|} = \frac{\left(\overrightarrow{x_{lead2}-x_{lead1}},\overrightarrow{y_{lead2}-y_{lead1}}\right)}{\sqrt{(x_{lead2}-x_{lead1})^2+(y_{lead2}-y_{lead1})^2}} = \|(\hat{x},\hat{y})\|\angle\theta \qquad (1)$$

$$\theta = \angle(\vec{x},\vec{y}) = \arctan\left(\frac{\vec{y}}{\vec{x}}\right), \qquad (2)$$

$(\hat{x},\hat{y})$ is the unit vector of the orientation. $(\vec{x},\vec{y})$ is the vector from the current checkpoint (x_{lead1},y_{lead1}) to the next (x_{lead2},y_{lead2}). $\|(x,y)\|$ is the magnitude of $(\vec{x},\vec{y})$, and θ is the angle of orientation as shown in Figure 1.

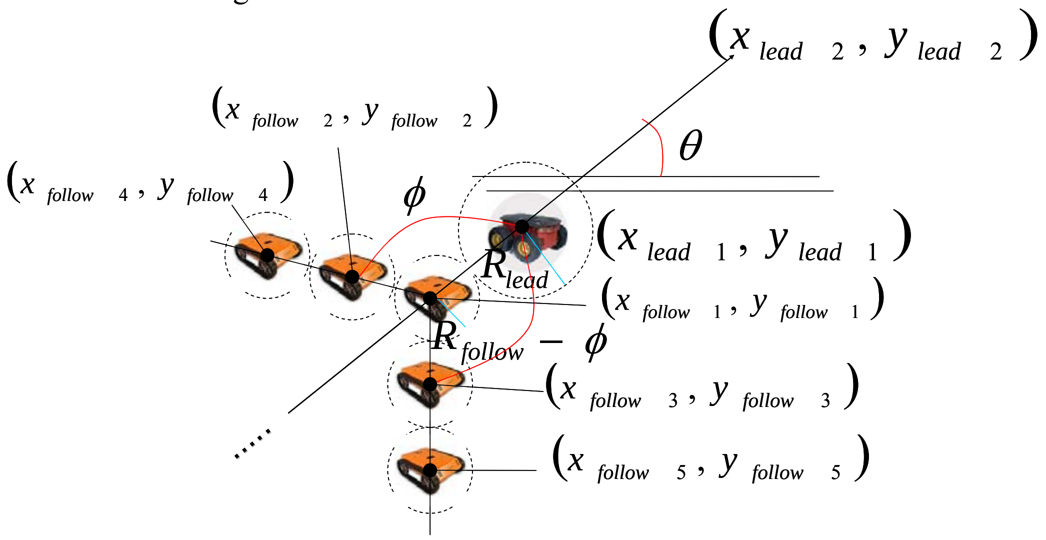
$(x_{lead\ 2}, y_{lead\ 2})$
$(x_{follow\ 2}, y_{follow\ 2})$
θ
$(x_{follow\ 4}, y_{follow\ 4})$
ϕ
$(x_{lead\ 1}, y_{lead\ 1})$
R_{lead}
$(x_{follow\ 1}, y_{follow\ 1})$
R_{follow}
$-\phi$
$(x_{follow\ 3}, y_{follow\ 3})$
$(x_{follow\ 5}, y_{follow\ 5})$

Figure 1. Relative positions between leader and followers in a V-formation are demonstrated in this figure. The leader is the P3-AT and the followers are the orange Traxsters.

The relative spacing between UGVs are approximated by adding their respective radii.

$$R_{lf} = R_{lead} + R_{follow} \qquad (3)$$

$$R_{ff} = R_{follow} + R_{follow} \qquad (4)$$

Setting the followers in formation was designed to be an iterative process so that changing the number of followers would not require complete revision of the algorithm. The first follower is stationed directly behind the leader, and tags *left* and *right* are assigned to this follower.

$$\left(x_{follow1}, y_{follow1}\right) = \left(x_{lead1}, y_{lead1}\right) - \left(\hat{x}, \hat{y}\right) \cdot R_{lf} \quad (5)$$

$$\left(x_{left}, y_{left}\right) = \left(x_{follow1}, y_{follow1}\right) \quad (6)$$

$$\left(x_{right}, y_{right}\right) = \left(x_{follow1}, y_{follow1}\right) \quad (7)$$

The leader then proceeds to alternate between assigning followers behind the *left* and *right* wings at angles ϕ and $-\phi$ respectively. Throughout this process, *left* and *right* are retagged onto the leftmost and rightmost followers in the formation respectively.

$$\left(x_{2n}, y_{2n}\right) = \left(x_{left}, y_{left}\right) + \left(\hat{x}\cos(\theta+\phi), \hat{y}\sin(\theta+\phi)\right) \cdot R_{ff} \quad (8)$$

$$\left(x_{2n+1}, y_{2n+1}\right) = \left(x_{right}, y_{right}\right) + \left(\hat{x}\cos(\theta-\phi), \hat{y}\sin(\theta-\phi)\right) \cdot R_{ff} \quad (9)$$

$$\left(x_{left}, y_{left}\right) = \left(x_{2n}, y_{2n}\right) \quad (10)$$

$$\left(x_{right}, y_{right}\right) = \left(x_{2n+1}, y_{2n+1}\right) \quad (11)$$

$$n = 1,2,3,\ldots \quad (12)$$

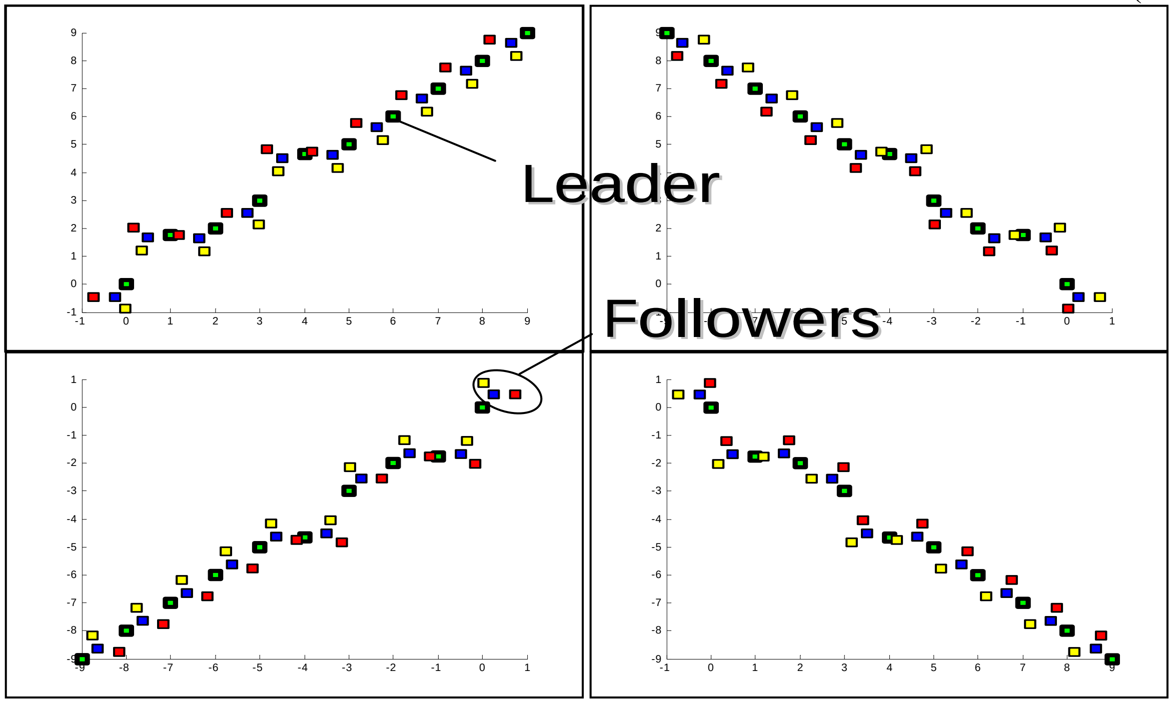

Figure 2 Simulations of trajectory planning of a complete UGV team consisting of 1 leader (black box) and 3 followers (blue, yellow, and red boxes) are shown in this figure. Four different trajectories are shown with the moving direction denoted as increasing x and increasing y (upper left), decreasing x and increasing y (upper right), decreasing x and y (bottom left), and increasing x and decreasing y (bottom right)

MATLAB simulations have been carried out. To simulate this V-shaped formation, a UGV team of a leader and three followers were set in MATLAB, and their respective trajectories were demonstrated in Figure 2. In these simulations, a vector of the leader's projected positions is first generated.[5,6,7] From here, the three follower's projected positions are calculated using the above equations (5 -12). These three vectors are then plotted with respect to time.

Collision-free strategy

If there is a potential collision with obstacles on the planned trajectory, the formation can be narrowed instead of correcting the trajectory. This maximizes the team's efficiency in reaching its destination, and allows the team to traverse through narrow doorways and spacing. After the obstacle is cleared, the team resumes its original formation. For simplicity, the followers will line up behind the leader in the order of (*follow*1, *follow* 2, *follow* 3,…) to narrow the span of the formation. The condition for lining up is dependent on the width of the formation and the spacing of the doorway,

$$W_{doorway} \geq W_s \tag{13}$$

where W_s is the width of the formation. This width is determined by the spacing between the rearmost followers (followers at point *s*),

$$W\text{s}=2\ \text{s}\ R_{ff}\ \sin(\pi-\phi). \tag{14}$$

For the simulations, the differences between widths of the formation and doorway as shown in Figure 3 are replaced with a narrow constant which reform the shape of the group. The greater W_s-$W_{doorway}$ is, the greater the narrow constant threshold.

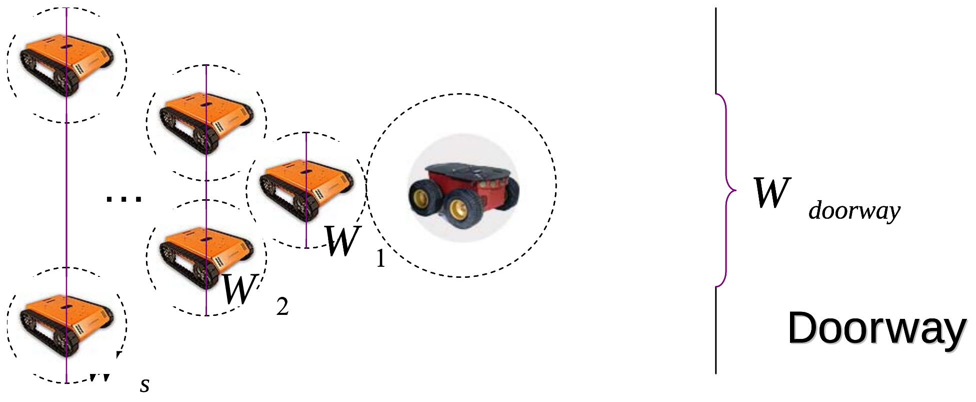

Figure 3. Widths of formation relative to a doorway obstacle are shown in this figure.

MATLAB simulations of the team traversing through doorways of different widths are presented in Figure 4.

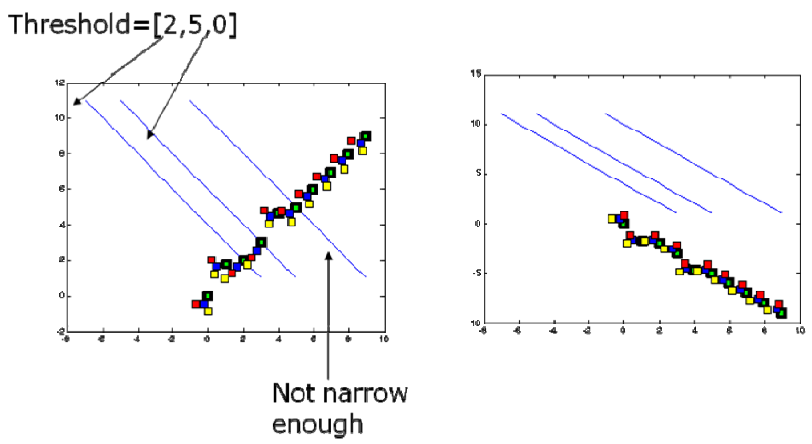

Figure 4. Simulations of team's trajectory response to doorway obstacles are demonstrated in this figure. Left: Traversing through doorways. Right: Avoiding doorways. Doorways are represented with blue lines. On the right, the team avoided doorways so there was no need to line up. On the left, the first and second doors' narrow thresholds were 2 and 5 respectively, signaling that the team needed to line up to fit through while the third door had a threshold of 0, signaling that the team can keep its current formation.

In these three lines with a parameter called threshold were set as shown in Figure 4. The width of the formation was calculated by taking the distance between the rearmost followers. If the threshold is greater than the width of the formation, the team's positions are recalculated so that the position of a follower is directly behind another. The order is predetermined, with the blue, red, and yellow. The second simulation demonstrates that no changes to the formation are necessary when no obstacles are traversed.

MATLAB simulations have also been performed with a number of different trajectories. A collision free path with two moving obstacles is shown in Figure 5 without considering the formation. Details of this algorithm can be found in our previous studies.[5,6,12]

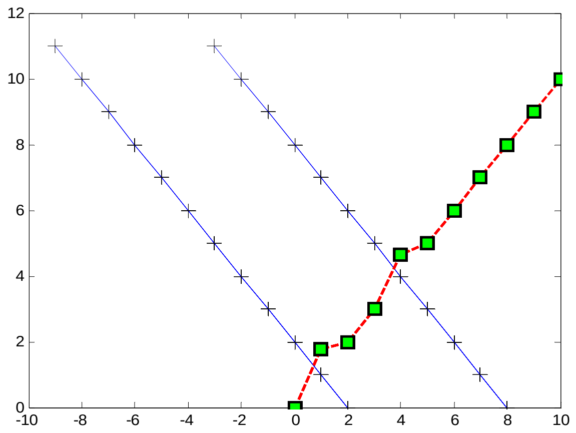

Figure 5. Collision free path of the leader (green box) starting from the position (0, 0) with two moving obstacle(cross) starting from positions (2, 0) and (6, 0) moving towards positions (-9, 11) and (-3, 11) respectively.

The "V" shape formation with sensed position of followers is shown in Figure 6. With the MATLAB simulation package, the UGV group could easily adapt to other formations for obstacle avoidance.

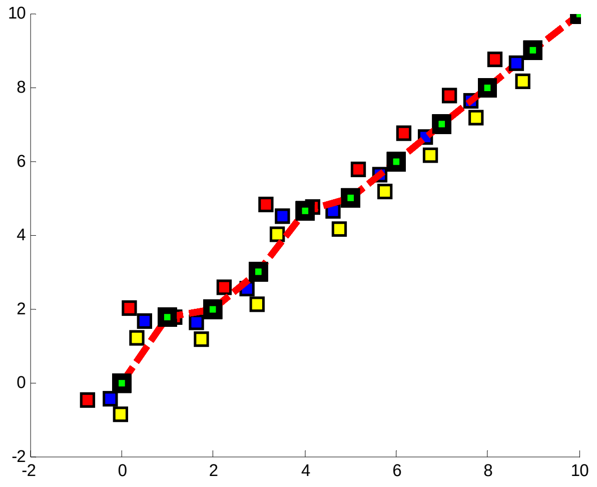

Figure 6. Formation control of the UGV group with two moving obstacles is demonstrated in this figure. The bold box filled with green denotes the leader. The red, blue and yellow boxes denote the followers.

Particle Swarm Optimization Controller

In order to improve the efficiency of the control scheme, the turn angle of followers will be minimized. To address this issue, PSO was introduced to the control the followers. The PSO algorithm developed for this study is as follows:

1. Divide interval between checkpoints into subintervals of equal length as shown in figure 7.

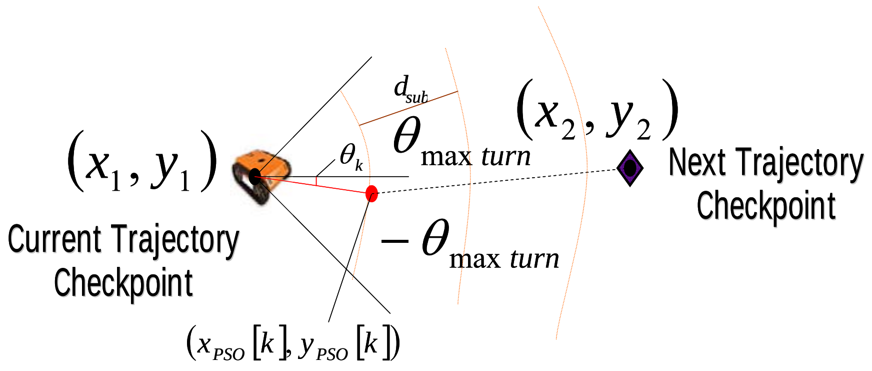

Figure 7. Trajectory division for PSO controller design

The distance of each interval is described as

$$d_{sub} = \frac{d[(x_1, y_1), (x_2, y_2)]}{n+1} = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{n+1} \quad (15)$$

2. Restrict the turn angle to a maximum and minimum based on the turning power of the follower.

$$-\theta_{\max turn} \le \theta_k \le \theta_{\max turn} \quad (16)$$

3. Spread an arbitrary number of random angles *k* within the range specified in step 2, and calculate x and y-coordinate values based on the current checkpoint *l* and these random angles as a random function. These are the particles of the PSO.

$$x_{PSO}(l,k) = lx_1 d_{sub} \cos\theta_k \quad (17)$$

$$y_{PSO}(l,k) = ly_1 d_{sub} \sin\theta_k \quad (18)$$

4. Calculate the distance from each random particle to the next checkpoint. These distances are the fitness functions of the particles.

$$f_{fitness}(l,k) = d[(x_{PSO}(l,k), y_{PSO}(l,k)), (x_2, y_2)] \quad (19)$$

5. Based on the fitness functions calculated in step 4, specify a local best $(\hat{x}(l), \hat{y}(l))$.

6. Move the other particles at the sub point toward the local best in hopes of finding a new local best.

$$(x_k, y_k) \xrightarrow{v_{update}} (\hat{x}, \hat{y}) \quad (20)$$

7. Repeat steps 1-6 for set number of swarm cycles

MATLAB simulations for this PSO controller design have been carried out and the simulation results are shown in Figure 8.

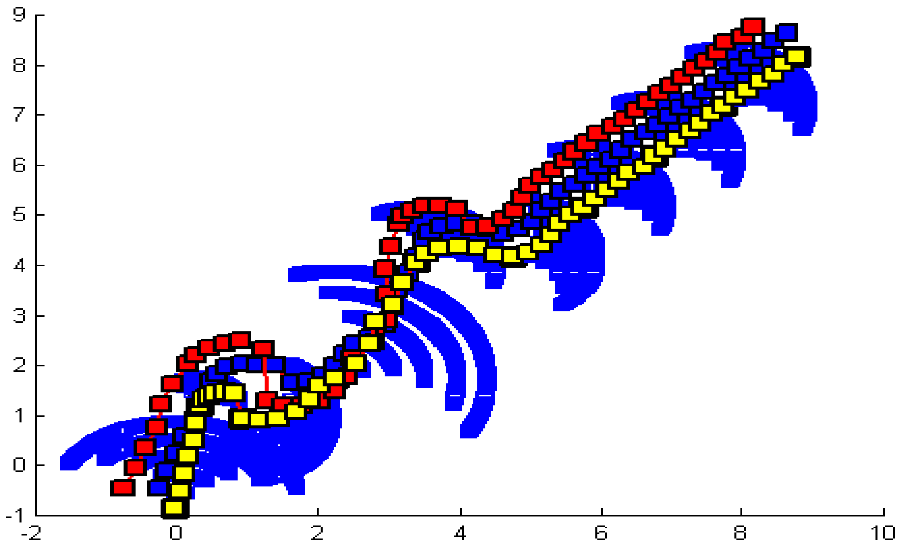

Figure 8. PSO controller with 5 subpoints, 50 random particles, π/16 max turn angle, and 5 swarm cycles. The red, blue, and yellow boxes demonstrated the optimized trajectory of the followers, each color representing a different follower. The blue arcs represent the particles. In between each checkpoint, 5 PSO iterations were implemented. These arcs become wider with every iteration indicating the widening range of turns the follower could make.

In this simulation, 50 random numbers were generated between the minimum and maximum turn angles. Based on these numbers, position points are calculated a set distance away from the current position, which are marked by the blue boxes. As the PSO progresses, these angles are updated so that the distances from the next set position is minimized while still adhering to the constraints set by the maximum turn angle. Theoretically, this should minimize the time the UGV would need to reach its next destination by minimizing the number of stationary turns. This simulation demonstrates that sharp turns can be mitigated when compared to only strictly path-planning in figures 4 through 6. Depending on the speed of the processor relative to the speed of the UGV itself, implementation of PSO itself may or may not be worth computation. Further investigation is required in determining the relative time complexities between computation speed and the speed of movement.

Conclusion

In conclusion, the design of a fully functional autonomous lead-follower formation was considered. The trajectory planning of the leader integrated into the follower position control and the formation control allows the UGV team to travel with a collision free path and keep a fixed formation. Also, implementation of a PSO into the system to improve path planning was demonstrated effectively in simulation. Future improvements to the system include the use of visual feedback to augment the formation control of the system and implementation of a rapid-response PSO to the real-platform. The application of these improvements may lead to optimal solutions discovery in minimal time and even more precise controllers may be developed.

References

1. K. Kobayahsi, Ka C. Cheok, K. Watanabe, and F. Munekata,“Accurate Differential Global Positioning System Via Fuzzy Logic Kalman Filter Sensor Fusion Technique.” *IEEE Transactions on Industrial Electronics*, Vol. 45, pp. 510-518. 1998.
2. C. Claus and C. Boutilier, “The dynamics of reinforcement learning in cooperative multiagent systems,” in *Proceeding of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin, July 1998, pp. 746–752.
3. L. Peshkin, K.-E. Kim, N. Meuleau, and L. P. Kaelbling, “Learning to cooperate via policy search,” in *Sixteenth Conference on Uncertainty in Artificial Intelligence*, San Francisco, 2000, pp. 307–314.
4. C. Boutilier, “Sequential optimality and coordination in multiagent systems,” in *Proceedings of the sixteenth International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, July/August 1999, pp. 478–485.
5. M. Bird, C. Khuc, E. Ortega, and C. Quiroz, “Lead-Follower Formation with Multiple Wirelessly Synchronized Autnomous Unmanned Vehicles.” August 2007.
6. Z. Qu, J. Wang, and C. E. Plaisted. “A New Analytical Solution to Mobile Robot Trajectory Generation in the Presence of Moving Obstacles,” *IEEE Transactions on Robotics*, vol. 20, pp. 978-993. *December 2004*.

7. Marcos Bird, Eric Ortega, Yufang Jin, “Lead-follower Control Scheme for Unmanned Ground Vehicles in an Unknown Environment”, Abstracts of IEEE Multi-conference on Systems and Control, Sep 2008, San Antonio, US.
8. M. Clerc and J. Kennedy, “The particle swarm-explosion, stability, and convergence in a multidimensional complex space,” *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 58–73, February 2002.
9. F. Chiang and R. Braun, “A nature inspired multi-agent framework for autonomic service management in ubiquitous computing environments,” in *Computational Intelligence Methods and Applications*, 2005.
10. N. Franken and A. P. Engelbrecht, “Particle swarm optimization approaches to coevolve strategies for the iterated prisoner’s dilemma,” *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 6, pp.562–579, December 2005.
11. H. S. Chang, “An adaptation of particle swarm optimization for markov decision processes,” in *IEEE International Conference on Systems,Man and Cybernetics*, vol. 2, 2004, pp. 1643–1648.
12. J. E. Marsden and A. J. Tromba. “The Geometry of Euclidean Space.” Vector Calculus, 5th Ed. New York: Freeman, 2003. pp. 1-38, 65-74.

SIYAO GU
Mr. Gu is currently a senior Electrical Engineering undergraduate student at Southern Illinois University. His research interests include advanced control algorithms.

MARCOS BIRD
Mr. Bird is currently pursuing a Master's in Electrical Engineering at the University of Texas at San Antonio. His research interests include inertial control systems and adaptive controllers.

SCOTT TIMME
Mr. Timme is currently a senior Electrical Engineering undergraduate student at the University of Texas at San Antonio. His research interests include advanced control systems for embedded devices.

ERIC ORTEGA
Mr. Ortega is currently a part time student pursing his Master’s in Electrical Engineering at the University of Texas at San Antonio.

YUFANG JIN
Dr. Jin currently serves as an Assistant Professor of Electrical Engineering at the University of Texas at San Antonio. Her research interests include robust adaptive control design for nonlinear systems, vision based control for mobile robots, synchronization and parameter estimation of chaotic systems, and observer design for nonlinear systems.