2005-8

Decentralized Perimeter Surveillance Using a Team of UAVs

Timothy McLain
Mechanical Engineering Department, Brigham Young University, mclain@byu.edu

Randal W. Beard
Department of Electrical Engineering, Brigham Young University, beard@ee.byu.edu

Derek Kingston
Brigham Young University - Provo

Ryan S. Holt
Brigham Young University - Provo

David W. Casbeer
Brigham Young University - Provo

Original Publication Citation

Kingston, D., Holt, R., Beard, R., McLain, T., and Casbeer, D. Decentralized Perimeter Surveillance Using a Team of UAVs, Proceedings of the AIAA Guidance, Navigation, and Control Conference, AIAA 2005-5831, August 2005, San Francisco, California.

Decentralized Perimeter Surveillance Using a Team of UAVs

Derek Kingston, Ryan Holt, Randal Beard[†] Timothy McLain, and David Casbeer

Brigham Young University, Provo, UT, 84602

This paper poses the cooperative perimeter surveillance problem and offers a decentralized solution that accounts for perimeter growth (expanding or contracting) and insertion/deletion of team members. By identifying and sharing the critical coordination information and by exploiting the known communication topology, only a small communication range is required for accurate performance. The basic perimeter surveillance solution is then extended to account for constrained turn radius. Finally, simulation results are presented that demonstrate the applicability of the solution.

I. Introduction

Perimeter surveillance has been a topic of interest for many years. Most researchers have focused on the sensor technology requisite for detection of a perimeter breach. Sensors that have been investigated include cameras,[1] ultrasound,[2] and radar.[3] The main focus seems to be on image processing techniques that use video feeds from multiple CCD cameras, but work has also been done using mobile robots[4] and underwater vehicles.[5] This paper will focus on the cooperative aspect of multiple unmanned air vehicles (UAVs) in regard to perimeter surveillance. Applications include border patrol, mobile combatant surveillance, and forest fire monitoring.[6,7]

In particular, the perimeters that we will focus on are large, possibly changing perimeters. We will assume that UAV agents have the proper sensor suite to detect changes in the perimeter and track the edge of the perimeter. We will not focus on the necessary sensor technology to do this, but rather on the algorithms that will allow a team of agents to monitor a perimeter in a decentralized fashion. Perimeter surveillance using multiple UAVs has the advantage of operating in a wide variety of circumstances such as changing perimeters (spill monitoring, forest fire surveillance) or very large perimeters (border patrol). In addition, small UAVs are relatively inexpensive and can be rapidly deployed.

Perhaps the most advanced example of perimeter surveillance with multiple agents is the MDARS project,[8] a joint effort between the Army and Navy. MDARS networks multiple ground robots to cooperatively monitor a fixed perimeter near critical storage facilities. Our work differs from MDARS in that we do not require team agents to be in constant communication with a centralized controller;[9] rather, agents are frequently outside of the communication range of the other team members and must in a decentralized manner monitor the perimeter.

One way to pose a cooperative control problem, such as decentralized perimeter surveillance, is in the framework of coordination variables.[10] The essential idea is to determine the key quantities which when all agents share a consistent view, cooperation is achieved. Section III presents a solution using the coordination variable approach to the basic perimeter surveillance problem of Section II. This basic solution is extended to account for constrained UAV turning radius in Section IV with simulation results presented in Section V. Finally, Section VI gives our conclusions.

II. Problem Formulation

The cooperative perimeter surveillance problem is to gather information about the state of the perimeter and transmit that data back to a central base station with as little delay and at the highest rate possible. For simplicity, consider point agents moving at a uniform constant velocity along a linear perimeter (see Figure 1). Agents can reverse direction instantaneously and always do so when the end of the perimeter is encountered. Communication between agents is only allowed when the agents are “touching”, i.e. when they occupy the same physical location. One way to visualize the problem is to imagine beads sliding along a string.

A configuration that allows information about the entire perimeter to be available at a high update rate and with low latency anywhere along the perimeter is for the agents to share, in equal portions, the length of the perimeter

[†]Corresponding author: beard@ee.byu.edu

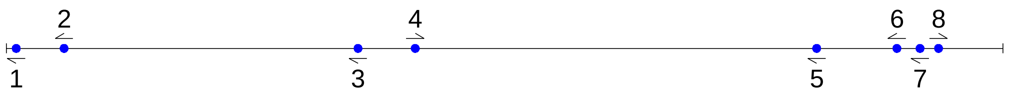
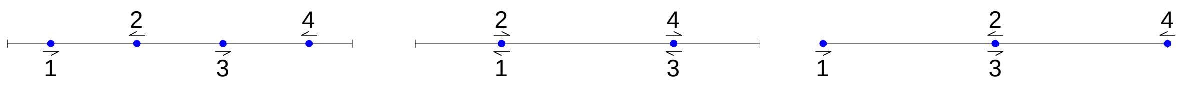


Figure 1. Example scenario where 8 agents monitor a linear perimeter.

while at the same time setting up an exchange pattern so that information flows to the ends of the perimeter. In other words, when agents are equally distributed along the perimeter and when each agent meets its neighbor at the end of the segment for which it is responsible, the ideal configuration has been reached.[6] Figure 2 shows the desired behavior for a team of four agents: the agents are uniformly distributed along the perimeter (Figure 2(a)) and each agent meets its neighbors at the end of its segments (Figures 2(b) and 2(c)). This oscillatory behavior of the agents requires that the team be synchronized not only in space (equally distributed), but also in time (meet neighbors at the end of segments). Note that any perimeter homeomorphic to a line will fit into the same framework and requires no new analysis.



(a) Agents are uniformly spread across the perimeter

(b) Neighbors meet and exchange perimeter state information

(c) Perimeter state is carried to all points along the perimeter

Figure 2. Information exchange pattern that allows information about the state of the perimeter to be available at any point along the perimeter.

III. Decentralized Solution

This section derives a decentralized algorithm to achieve the behavior described in Section II. The coordination variables for this problem are: (1) the perimeter length, (2) the number of agents on the left side of the perimeter relative to a given agent, and (3) the number of agents on the right side of the perimeter relative to a given agent. When each agent has consistent values for the coordination variables (corresponding to a given scenario) then each will be able to compute the perimeter segment for which it is responsible. The first step in the decentralized solution is to ensure that when each agent has the proper values, that coordination will be achieved.

The following algorithm ensures that if each agent has coordination variable values consistent with the current scenario (i.e. each agents knows the length of the perimeter, the total number of agents on the team, and its position in the team), then the desired steady state behavior will reached.

Algorithm 1: Distributed spread

```
if rendezvous with neighbor then
    calculate shared border position
    travel with neighbor to shared border
    set direction to monitor own segment
else if reached perimeter endpoint then
    reverse direction
else
    continue in current direction
```

For every consecutive pair of agents, there is a single position where their two segments border. When each agent has a consistent knowledge of the length of the perimeter and its order in the team, then the endpoints of its responsible segment are easily computed. The endpoint shared with a neighbor is the shared border position to which both will travel together in the first phase of Algorithm 1. Effectively, each agent escorts its neighbors to the position at which they should have met had they been in perfect synchronization. Since agents only reverse direction at perimeter ends and when they meet other agents, each agent is guaranteed to meet its neighbors.

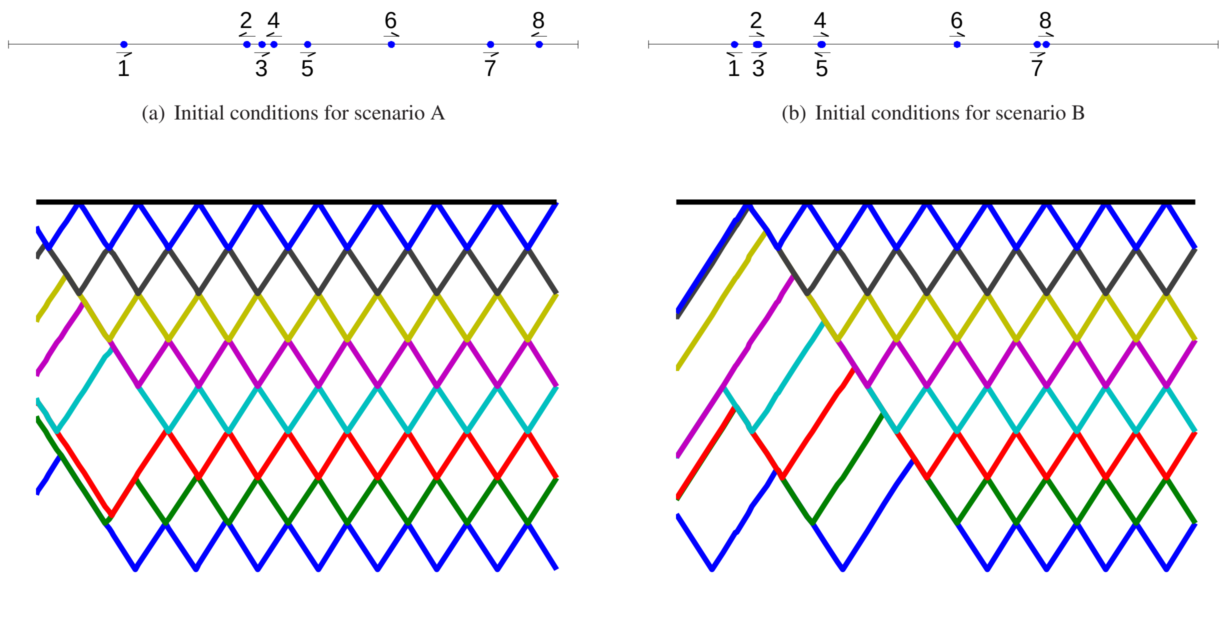
Theorem 1. *Let the perimeter length and number of agents be fixed. If all agents have coordination values consistent with the scenario, then Algorithm 1 ensures that the desired steady-state behavior is achieved after time $2T$ has passed where T corresponds to the time required for one agent to travel the length of the perimeter.*

Proof. In general, the team agents can initially be positioned anywhere along the perimeter and be traveling either in the positive or negative direction (recall that constant uniform velocity is assumed). Since each agent has a consistent understanding of the coordination variables, then each can calculate the segment along the perimeter for which it is responsible. As is noted above, agents are guaranteed to meet both neighbors since Algorithm 1 only commands agents to reverse direction at a perimeter endpoint or in order to advance to its segment boundaries.

For N agents, order the segments from the left edge of the perimeter $1, \ldots, N$ and let the number of each agent correspond to the segment for which it is responsible. Consider the effect of Algorithm 1 on the leftmost agent (agent 1). Once agent 1 has escorted its right neighbor to their shared border, then no agent to the right of agent 1 will ever travel along segment 1 again. This can be seen by noting that after agents 1 and 2 split at their shared boundary *both* will travel the length of one segment to get to the opposite end of their respective segments. If agent 2 meets agent 3 along the way, then agents 2 and 3 will continue to their shared border before agent 2 reverses direction and so, agent 2 will travel at least one segment length away from the boundary between segments 1 and 2. Since both travel at a uniform constant velocity, then agent 1 will arrive back at the border between segments 1 and 2 at the same time or before agent 2, but never after. Now consider agent 2 after being escorted by agent 1 to their shared boundary. Since by this time agent 2 never ventures into segment 1, the border between agents 1 and 2 can be regarded as a fixed perimeter endpoint for agent 2. In other words, the same analysis now holds if we consider agent 2 the leftmost agent in a set of $N - 1$ agents. Observe that the same argument holds starting with the rightmost agent and considering all agents to the left. Therefore, there is a time τ after which all agents are only found on their respective segments. This implies that the desired steady-state behavior of Section II has been achieved.

The worst case situation occurs when all agents are stacked infinitesimally close at one end of the perimeter and are traveling toward the other. Once T has passed all agents are at the opposite end of the perimeter where they meet both neighbors. Each pair will travel to their shared borders which for the farthest pair will require a travel time less than T. Therefore, the steady-state behavior will be achieved before time $2T$. $\square$

Figure 3 shows two simple scenarios with 8 agents spreading out over a fixed perimeter where each agent begins with a consistent understanding of the coordination variables. The lattice geometry indicates that the desired steady-state behavior has been reached. Note that the agents require very few meetings with other agents to converge to the proper configuration.



(a) Initial conditions for scenario A

(b) Initial conditions for scenario B

(c) Position of agents in scenario A

(d) Position of agents in scenario B

Figure 3. Example scenarios for point agents whose spread is governed by Algorithm 1

Theorem 1 ensures finite time convergence to the steady-state behavior of Section II when the coordination variables are consistent among the team. By adding the ability to update the coordination variables, Algorithm 1 can be modified to ensure that there will be a time when the coordination variables are consistent. This is done by simply incorporating local variables that track the amount of perimeter distance and number of agents on both sides. These are easily updated when meeting with another agent on the team – both agents have knowledge about the number of agents and perimeter length on their respective sides. If the perimeter and number of agents is fixed, then the coordination variables will eventually be consistent among the team since agents are guaranteed to meet both neighbors. Once the coordination variables are consistent, Theorem 1 ensures that the desired steady-state behavior will be achieved. Note that the same method used to update the coordination variables can also be used to detect changes in the perimeter or insertion/deletion of team members. In other words, the core ideas from Algorithm 1 can be used to monitor step changes in perimeter and team size in a decentralized manner.

By modifying Algorithm 1 to allow agents to transmit information regarding the perimeter size and number agents, changes in perimeter size can be tracked. Figure 4 shows agent tracking on a perimeter with a step change in size and a perimeter with sinusoidal growth. Recall that the modifications to Algorithm 1 make the team robust to step changes in perimeter size, but the same algorithm also allows good tracking for other types of perimeter growth. Note that each agent has no knowledge *a priori* of the perimeter length or number of agents on the team. These coordination variables are updated through repeated interactions with other team members.

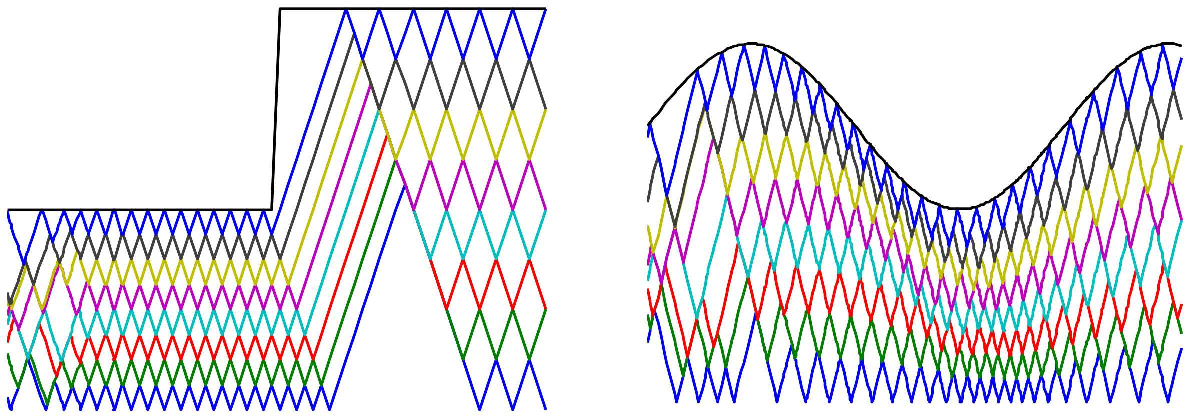

(a) Positions of agents for step change in perimeter (b) Positions of agents for sinusoidal perimeter

Figure 4. Point agents tracking changing perimeters using a modified version of Algorithm 1 to continuously update the coordination variables. Agents learn the size of the perimeter and number on the team through repeated interaction with other team members.

IV. Extension to UAVs

Since UAVs have a constrained turning radius, the assumption of instantaneous turn-around time of Section III is unrealistic. The purpose of this section is to investigate the limitations of Algorithm 1 when applied to perimeter surveillance using a team of UAVs.

In Section II agents were points that could communicate only when touching. Now consider UAVs flying at constant velocity with turning radius R (simple Dubin's car model). To complete a U-turn with such a turning radius requires that the UAV be aware of the need to turn around a distance of $\frac{7}{6}\pi R$ before the actual rendezvous.[11] Since both UAVs must have a buffer of this distance in order to complete a turn-around, the communication radius of each UAV must be greater than $\frac{7}{3}\pi R$.

The constrained turning radius also affects how many agents can monitor a perimeter without being interrupted during a turn. The smallest segment is one on which a UAV can complete two U-turns consecutively, so a perimeter of length P being monitored by UAVs with radius R can have at most $\lfloor \frac{3}{7\pi R} P \rfloor$ agents.

Consider a scenario with N UAV agents that have a large enough communication radius to perform the U-turn maneuver as described above. Also, let the perimeter be large enough so that in steady-state the segment for which a UAV is responsible is large compared to the turning radius of the vehicle (specifically, larger than $\frac{7}{3}\pi R$). Under what conditions can we ensure that during the transient period, no UAV is interrupted while performing a turning maneuver? In other words, when is the distance between consecutive direction reversals smaller than the distance required to perform those maneuvers?

It is easy to show that when the scenario allows for uninterrupted steady-state behavior and the agents have consistent coordination variables, that no UAV is interrupted during the transient period. However, during the time when the team is forming a consistent set of coordination variables, the distance between consecutive direction changes can be made arbitrarily short with proper choice of initial positions and directions of the agents. A sufficient condition to ensure that no UAV is interrupted in the transient period is to enforce a separation distance of at least $\frac{7}{3}\pi R$ at launch. In the general case, Algorithm 1 must be further modified to allow interrupted turns to translate into effective perimeter growth. These effects will die out as soon as the coordination variables reach consistency.

V. Simulation Results

To verify the feasibility of implementing Algorithm 1 on a team of small UAVs, a high fidelity simulation was performed. The simulation scenario involves four UAVs monitoring a perimeter composed of 12 waypoints with total length of 3.1 km. The physics of each UAV are determined through full nonlinear 6 degree-of-freedom equations of motion.[12] In addition, each virtual UAV is equipped with autopilot software that enables accurate waypoint tracking[13] with turning radius of approximately 50 meters. The communication model allows UAVs to communicate only to adjacent neighbors who are inside the communication range, approximately 370 meters (the minimum necessary).

The scenario starts with 3 of the 4 UAVs being launched in rapid succession. Each starts with no knowledge of the number of agents on the team or the perimeter length (even though the perimeter is defined by predetermined waypoints, we require the UAVs to treat the perimeter length as initially unknown). Later, the fourth UAV is launched to test the affect of insertion. Figure 5 shows the parameterized position of each UAV along the length of the perimeter. Note that in the regions where the team should be locked into the ideal configuration, some position overlap is observed. This is due to the inability of the UAVs to perform the U-turn maneuver precisely and acts as a disturbance to the system. The overall behavior of the team is as expected with the team reaching the desired steady-state behavior quickly and reacting to a step in team size.

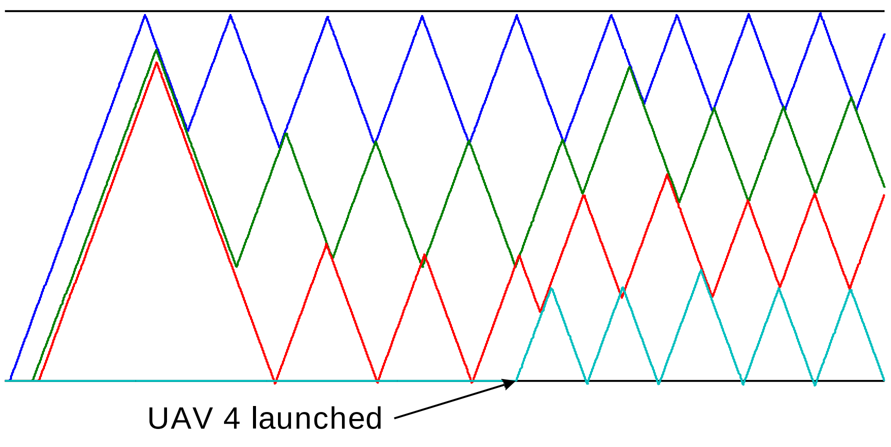


Figure 5. Parameterized position of each UAV along the perimeter.

VI. Conclusions

This paper has presented a decentralized algorithm for perimeter surveillance that converges in finite time. By sharing information regarding the perimeter length and number of team members, each agent obtains a consistent set of coordination variables that allows the decentralized algorithm to operate. Advantages include the ability to monitor changing perimeters, account for dynamic insertion and deletion of team members, and operate with small communication range in a decentralized manner. Extension of the algorithm to account for UAV turn radius constraints was presented along with simulation results.

Acknowledgements

This research was supported by NASA under STTR contract No. NNA04AA19C to Scientific Systems Company, Inc (SSCI) and Brigham Young University (BYU), by the National Science Foundation under Information Technology Research Grant CCR-0313056, and by the United States Air Force under AFOSR Award number FA9550-04-1-0209. Special thanks to David Johansen for help with flight testing software.

References

[1]Young, S., Forshaw, M., and Hodgetts, M., "Image comparison methods for perimeter surveillance," *Proceedings of the International Conference on Image Processing and Its Applications*, 1999.

[2]Peralta, J. O. and de Peralta, M. T. C., "Security PIDS with physical sensors, real-time pattern recognition, and continuous patrol," *IEEE Transactions on Systems, Man and Cybernetics, Part C*, Vol. 32, Nov. 2002, pp. 340–346.

[3]Barry, A. S. and Czechanski, J., "Ground surveillance radar for perimeter intrusion detection," *Proceedings of the Digital Avionics Systems Conference*, 2000.

[4]Everett, H. R., "Robotic security systems," *IEEE Instrumentation & Measurement Magazine*, Vol. 6, No. 4, Dec. 2003, pp. 30–34.

[5]Kemp, M., Bertozzi, A. L., and Marthaler, D., "Multi-UUV perimeter surveillance," *Proceedings of the IEEE/OES Autonomous Underwater Vehicles Conference*, 2004.

[6]Casbeer, D. W., Kingston, D. B., Beard, R. W., McLain, T. W., Li, S.-M., and Mehra, R., "Cooperative Forest Fire Surveillance Using a Team of Small Unmanned Air Vehicles," *International Journal of System Sciences*, accepted April 2005, Technical Report available at https://dspace.byu.edu/handle/1877/55.

[7]Casbeer, D. W., Li, S.-M., Beard, R. W., McLain, T. W., and Mehra, R. K., "Forest Fire Monitoring Using Multiple Small UAVs," *Proceedings of the American Control Conference*, 2005.

[8]Space and Naval Warfare Systems Command, "Mobile Detection Assessment and Response System (MDARS)," http://www.nosc.mil/robots/land/mdars/mdars.html.

[9]Laird, R. T., Everett, H. R., Gilbreath, G. A., Heath-Pastore, T. A., and Inderieden, R. S., "MDARS Multiple Robot Host Architecture," *Association of Unmanned Vehicle Systems, 22nd Annual Technical Symposium and Exhibition*, 1995, Available at http://www.nosc.mil/robots/land/mdars/auvsmrha.html.

[10]Ren, W., Beard, R. W., and McLain, T. W., *Cooperative Control*, Vol. 309, chap. Coordination Variables and Consensus Building in Multiple Vehicle Systems, Springer-Verlag Series: Lecture Notes in Control and Information Sciences, 2004, pp. 171–188.

[11]Kingston, D. B., *Implementation Issues of Real-Time Trajectory Generation*, Master's thesis, Brigham Young University, 2004, http://contentdm.lib.byu.edu/ETD/image/etd357.pdf.

[12]Stevens, B. L. and Lewis, F. L., *Aircraft Control and Simulation, 2nd Edition*, John Wiley & Sons, Inc., 2003.

[13]Beard, R., Kingston, D., Quigley, M., Snyder, D., Christiansen, R., Johnson, W., Mclain, T., and Goodrich, M., "Autonomous Vehicle Technologies for Small Fixed Wing UAVs," *AIAA Journal of Aerospace Computing, Information, and Communication*, (to appear).