Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Distributed Balancing of AAVs for Uniform Surveillance Coverage

Brandon J. Moore and Kevin M. Passino*
Dept. Electrical and Computer Engineering
The Ohio State University
Columbus, OH 43210
mooreb@ece.osu.edu, passino@ece.osu.edu

Abstract— **This paper addresses the problem of enabling a group of autonomous air vehicles to provide surveillance coverage for an area significantly larger than their communication radius. Our formulation spatially decomposes the overall surveillance mission into subtasks and we develop a distributed cooperative control algorithm that continuously reassigns AAVs to these subtasks based on only local information in order to achieve the most balanced distribution of AAVs possible within a finite time interval of known bound. Various applications are discussed and simulations are included to illustrate convergence dynamics as well as to measure practical performance as a function of problem parameters.**

I. Introduction and Problem Statement

This work is motivated by a mission scenario in which a group of autonomous air vehicles (AAVs) with proximity restricted communications are tasked to provide cooperative surveillance for a large area of interest. If the mission is simply to find stationary targets, the AAVs could simply fly in a formation that preserved communication connectivity and sweep the area together in order to find those targets (e.g. the "mowing the lawn" approach taken in [1]). However, we are more often concerned with targets that are either mobile or "pop-up" in nature [2], [3], and can easily evade detection by a single formation of AAVs. Thus our primary desire is to disperse the AAVs throughout the area of interest as uniformly as possible in order to maximize the probability that a particular target will be spotted by at least one AAV. Because this dispersal means that the AAVs will be out of communication range of each other most of the time, we need to develop a way for the group to detect and react to events that destroy the uniformity of their distribution, such as the possible loss of AAVs due to attack, refueling needs, mechanical failures, and reassignment to other tasks, or the possible deployment of extra AAVs to their group.

In this paper we consider a surveillance mission that is composed of many subtasks; we partition the whole region to be covered into smaller *areas* to which the individual AAVs are then assigned in order to provide visibility for a particular part of the environment (i.e. an AAV assigned to a certain area will limit its search path to that area rather than exploring the whole environment). In order to provide distributed control for the group of AAVs, each area will be presided over by one or more *coordinators* who direct the actions of the individual AAV. In practice these coordinators would most likely be another AAV (of a special type perhaps) that would loiter in a certain location by flying a predetermined route along the boundary between two or more adjacent areas. A simple example illustrating this concept appears in Figure 1. The coordinators make decisions concerning the movement of AAVs from one area to another, creating a hierarchical structure in which the surveillance AAVs, while possibly having a large degree of autonomy in regard to how they conduct their search within the area to which they are assigned, essentially become resources that the coordinators may distribute among the different areas according to higher level considerations. That is to say that a coordinator makes local decisions to transfer AAVs between two of the areas it "connects" by virtue of its physical location and that this decision is based on the need for surveillance in the different areas. We can see from the above description that the overall system can be organized as a graph; if we think of the areas as the vertices of this graph, then we can consider each coordinator to be an edge (or set of edges) because it defines a link between the areas it connects. The surveillance AAVs can then be moved around this graph to achieve the desired distribution.

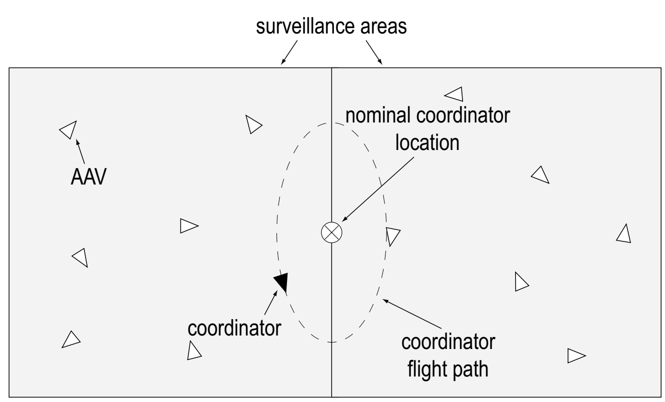

Fig. 1. A simple two surveillance area example.

By itself, the partition of the whole region into smaller areas may not by itself solve the communication problem. As previously mentioned, communication between the AAVs may be limited by a very small transmission radius due to such things as hardware shortcomings (e.g. a low-power transmitter on a micro-AAV) or by special operational re-

*This work was supported by the AFRL/VA and AFOSR Collaborative Center of Control Science (Grant F33615-01-2-3154). In addition, the work was supported by a DAGSI Fellowship to B. Moore.

quirements (e.g. the voluntary limitation of the strength and duration of radio transmissions for stealth missions). When this is the case, the surveillance AAVs would have to "check in" periodically with each coordinator for the area to which it is currently assigned by flying to where those coordinators are loitering. These periodic visits accomplish a couple of things. Since the coordinators cannot always communicate directly with the AAVs or the other coordinators, they can only estimate how many AAVs are in the area at any given point in time. When a new AAV is added to the area by one coordinator, it must eventually visit the other coordinators for that area so that they can update their estimates. By extension, if a limit is imposed on the maximum length of time an AAV is allowed to go between visits to any given coordinator, then if a coordinator has not seen an AAV for that amount of time it can safely assume that it has been removed from the area and can adjust its estimate accordingly. In this way it can be said that each coordinator senses the number of AAVs in an area but with some inaccuracy due to the delays involved. The other thing these periodic visits accomplish is to give a coordinator a period of time in which they have sole control over an AAV and can transfer it to another area without having to worry about other coordinators trying to transfer the same AAV at the same time.

In regards to the distribution of the surveillance AAVs, we have already stated that our objective is to disperse them as uniformly as possible throughout the region of interest. In terms of the framework we have been discussing this means that we would like each area to have a number of AAVs proportional to its physical size (so that the density of AAVs in each area is as equal as possible). However, if we have special information that makes surveillance a higher priority in one part of the region than another, then this is not necessarily the case. In order to handle a wider range of problem scenarios while keeping our formulation simple, let us instead make it our objective to distribute the AAVs as evenly as possible among the areas and assume that the region of interest is partitioned in a manner such that the size of an individual area is inversely proportional to the priority of the area it covers. If we then want a uniform density of AAVs we can partition the region along a regular grid with areas of equal size (see Figure 2), but if we want greater surveillance in a particular section of the region we can design the areas that cover that section to be smaller than the rest so that the density of AAVs is greater there (see Figure 3).

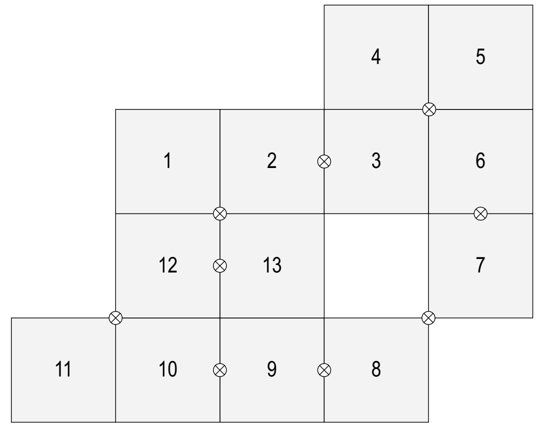

Fig. 2. Example of a surveillance area layout with uniformly sized areas. Shaded squares are surveillance areas and white crossed circles are nominal coordinator locations (a coordinator connects all areas this circle touches).

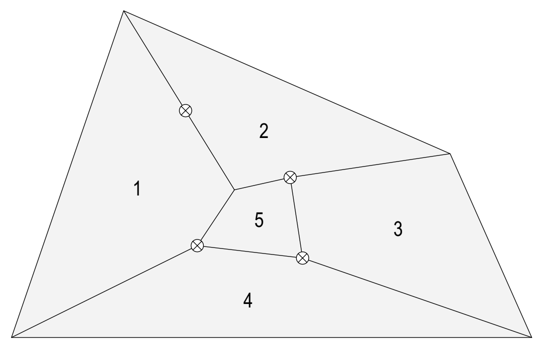

Fig. 3. Example of surveillance area layout with areas of unequal size. The density of AAVs in area 5 will be higher than the other areas if the number of AAVs per area is the same.

The balancing of surveillance AAV resources is conceptually related to traditional load balancing objectives in distributed computing [4] (with the AAVs viewed as a finite number of discrete load blocks and the areas as the processors) with a few key exceptions. In this scenario, decisions to move AAVs from a given surveillance area to a neighboring one are made in a distributed manner at the interconnections between the two areas (i.e. by the coordinators) rather than coordinated centrally as they are in a system where a single processor controls an area. Also, in discrete load balancing systems, a locally balanced state is all that is guaranteed, so neighboring processors may differ by as much as the size of the largest load block for both the delay and non-delay cases [5], [6] and so the global imbalance grows with the diameter of the network. While such imbalances may be acceptable in a system with a large total load, since the ratio of AAVs to areas may be fairly small in these missions it is important to achieve a globally balanced state. That is to say that each surveillance area should have at most one AAV more than any other area. The message passing algorithm presented in [7] is able to achieve a state in which every processor's set of neighbors is balanced within the largest block size, but is still not able to guarantee that that will result in a globally balanced state. It is also possible to reduce the expected size of the global imbalance by employing nondeterministic algorithms such as the one presented in [8]. A final difference between the usual load balancing framework and our problem is that in traditional load balancing systems there is the desire that the occurrence and volume of load transfer events should diminish as load imbalances become small. We, however, might prefer that whenever the number of AAVs is not evenly divisible by the number of areas, that the "excess" AAVs continue to transition between different areas so that some sort of average coverage is achieved on each one.

We note that after submission of this paper we found work in [9] that contains ideas similar to those in the algorithm we develop here. That work was done in a highly specialized context of load balancing for a grid of parallel processors and does not address the delays and asynchronicity present in our model.

II. AAV Balancing on a Unidirectional Ring of Areas

The problem introduced in Section I may prove very difficult to solve for an arbitrary graph; the physical delays associated with AAVs traveling between coordinators is likely to result in unstable oscillations in the system because the coordinators are forced to make decisions based on outdated information. In both [4] and [5], the distorting effects of delays are overcome by algorithms that ensure "bad" information is eventually purged from the system. We accomplish the same effect in this work by restricting the interconnection graph of the areas to a special structure. Each surveillance area will be connected to exactly two others and instead of allowing the connecting coordinators to transfer AAVs in either direction we stipulate that on each surveillance area one coordinator may only add AAVs to that area while the other one may only remove them. Since we implicitly assume the graph of areas is connected, the structure we are discussing is a unidirectional ring. We note that focusing on this topology is not as restrictive as it might first appear. For instance, the surveillance area layouts in both Figures 2 and 3 can both be converted to unidirectional rings by simply restricting the action of the coordinators. The only layouts that cannot be converted in this manner are those whose underlying graph does not contain a Hamiltonian cycle (e.g. a line graph or a tree), but so long as the surveillance area layout is a matter of design these can be avoided.

For convenience, let us associate each surveillance area with the coordinator that removes AAVs from it and refer to this combination as a *node*. If there are N areas in our system, let us denote the set of nodes by $\mathcal{N} = \{1, \ldots, N\}$. The graph $(\mathcal{N}, \mathcal{A})$, with $\mathcal{A} = \{(i, i+1) : i, j \in \mathcal{N}\}$ (with $i+1$ denoting modulo N addition throughout this paper whenever $i \in \mathcal{N}$) then describes our AAV balancing network. Specifically, the coordinator of node i senses the AAVs on the areas of nodes i and $i+1$ and may transfer them from node i to (and only to) node $i+1$.

The manner in which a coordinator senses the AAVs of its node and those of another is subject to substantial delays. A coordinator can only directly detect the presence of an AAV on a surveillance area when it flies near the loitering location of the coordinator; it has to estimate the total number AAVs on the areas to which it is connected based on what it knows about those areas and the dynamics of the AAVs. Specifically, a coordinator will only notice that an AAV has been added to a surveillance area when it flies near that coordinator's location. Since the areas are finite we can assume that when an AAV is added to a node, the node's coordinator will notice the new AAV within some time delay that is finite and has a known bound. Similarly, once a coordinator has recognized that an AAV is on a surveillance area, if that AAV is later removed, the coordinator will recognize that it is gone within some bounded time interval. Initially, the actual number and location of the AAVs in the system will generally be different from each coordinator's estimate, with each coordinator thinking that there are some AAVs on a surveillance area when they are not and not knowing about some AAVs that are there. If the total number of AAVs in the system remains constant after a certain time, however, the coordinators should be able to intelligently update their estimate of AAV numbers so that they eventually have an estimate that, while not completely accurate, at least possesses some relationship to the real numbers that allows the system to achieve a globally balanced state.

A. Model

We will model our AAV balancing network with a discrete event system model of the same form used in [5]. The state of the system will be updated at discrete time instants according to the occurrence of particular events describing the relevant behavior of the system. The sequence of events is non-deterministic but is constrained by the state of the system at each time step as well as by casual links between pairs of events over time.

The state of the complete system will be composed of those state variables describing the plant (the location of AAVs) and those describing the distributed controller (the coordinator's estimates of the AAVs locations). For the plant, each node $i \in \mathcal{N}$ has an associated *resource level* for its surveillance area equal to the number of AAVs on that node's surveillance area and denoted as $x_i \in \mathbb{N}$, where we take the definition of the natural numbers to be $\mathbb{N} = \{0, 1, 2, \ldots\}$ (i.e. the non-negative integers as opposed to just the positive integers as is sometimes seen). Let $x_P = [x_1, \ldots, x_N]^\top$ be the entire state of the plant. For the controller, the coordinator of node i has a (possibly outdated) estimate of the true values of its own resource level and that of node $i+1$. Let these values be denoted by $\tilde{x}_i^i$ and $\tilde{x}_i^{i+1}$, respectively, and let $x_C = [\tilde{x}_1^1, \tilde{x}_1^2, \ldots, \tilde{x}_N^N, \tilde{x}_N^1]^\top$ denote the entire state of the controller. The state of the system is then $x = [x_P^\top, x_C^\top]^\top$ and we let the state space be denoted by $\mathcal{X} = \mathbb{N}^{3N}$. Let $x(k) \in \mathcal{X}$ denote the state of the system at logical time index k (with similar notation for its components).

There are three subsets of this state space in which we are interested. The first,

$$\mathcal{X}_L = \left\{ x \in \mathcal{X} : \sum_{i=1}^{N} x_i = L \right\}$$

with $L = \sum_{i=1}^{N} x_i(0)$, is actually a family of subsets parameterized by the initial condition $x_P(0)$. One of our primary assumptions will be that no AAVs enter or leave the system during its operation, which is a standard assumption in load balancing formulations [4], [5]. This assumption is equivalent to the statement that $x(k) \in \mathcal{X}_L$ for all $k \geq 0$.

The second subset of interest,

$$\mathcal{X}_E = \left\{ x \in \mathcal{X} : \tilde{x}_i^i \leq x_i \leq \tilde{x}_{i-1}^i \ \forall \ i \in \mathcal{N} \right\}$$

defines a condition where the coordinator for each node i does not overestimate the resource level of the surveillance area of node i nor underestimate the resource level of the surveillance area of node $i+1$. By assuming that $x(k) \in \mathcal{X}_E$ for all $k \geq 0$, we will be able to greatly simplify our model and later analysis. This assumption requires some justification, however, because an arbitrary initial state for the controller $x_C(0)$ is not guaranteed to posses any relationship to the real resource levels $x_P(0)$.

Take an arbitrary initial condition for the system and consider the coordinator of node i. This coordinator's estimate of the number of AAVs on the surveillance area of its node is based on 1) some AAVs it thinks are present and are, 2) some AAVs it thinks are present but are not, and 3) a number of AAVs that are present but the coordinator has not yet detected. Based on the description of the system above, there is a limited amount of time that can pass before the coordinator will recognize the absence of all the AAVs it initially believed to be present. Since that coordinator is the only one which can remove AAVs from its own surveillance area (and it keeps track of those), after some time period the coordinator's estimate for that area differs from the real number of AAVs only by those AAVs which are present but have not yet been detected (and hence $\tilde{x}_i^i \leq x_i$). Similarly, since the same coordinator is the only one that can add AAVs to the surveillance area of node $i+1$, after a finite time that coordinator will have detected every AAV on that area, but possibly believe that some AAVs removed by the coordinator of node $i+1$ are still present, and so its estimate of the number of AAVs on that surveillance area will be no less than the actual value (i.e. $\tilde{x}_i^{i+1} \geq x_{i+1}$). Since the previous argument holds regardless of the initial condition of the system and because the state $x_P(k)$ is bounded for all k (because $x(k) \in \mathcal{X}_L$ for $k \geq 0$), we can, without any loss of generality, assume that the state of the system lies within the set $\mathcal{X}_E$ for all time. For convenience, let $\mathcal{X}_0 = \mathcal{X}_L \cap \mathcal{X}_E$ be the intersection of the these first two subsets, giving us $x(k) \in \mathcal{X}_0$ for all k.

The last subset of interest is the goal set that we want the system to reach (and remain within). This set is given by

$$\mathcal{X}_I = \{ x \in \mathcal{X}_0 : |x_i - x_j| \leq 1 \text{ for all } i, j \in \mathcal{N} \}$$

and represents the set of all states lying in $\mathcal{X}_0$ such that resource levels of all nodes are balanced to within one unit of each other. This is the best configuration possible given that $x_P \in \mathbb{N}^N$ and the fact that the total number of AAVs may not be divisible by N.

There are two types of events of interest in this system, namely, the passing of AAVs from one node to another and a change in a coordinator's resource level estimates (when it recognizes that an AAV has been added to or removed from a surveillance area). Let $e_i^{\alpha(i,k)}$ denote the "partial" event of node i passing $\alpha(i,k) \in \mathbb{N}$ AAVs to node $i+1$ at time k. By the logic underlying our assumption that the initial state lies in $\mathcal{X}_0$, we need only consider estimate changes for node i's coordinator that either increase its estimate of the resource level of its own surveillance area or decrease its estimate of the resource level of the surveillance area of node $i+1$. Therefore, let $e_i^{\beta(i,k,m)}$ denote that at time k the coordinator of node i has detected $\beta(i,k,m) \in \mathbb{N}$ of the AAVs that were added to its own surveillance area at time $m \leq k$. Similarly, let $e_i^{\gamma(i,k,m)}$ denote that at time k the coordinator of node i has noticed the absence of $\gamma(i,k,m) \in \mathbb{N}$ of the AAVs that were removed from the surveillance area of node $i+1$ at time $m \leq k$.

Let the total event space $\mathcal{E}$ be defined as the union of the following sets minus the null set $\emptyset$,

$$\mathcal{P}(\{e_i^{\alpha(i,k)} : i \in \mathcal{N} \text{ and } \alpha(i,k) \in \mathbb{N}\})$$

$$\mathcal{P}(\{e_i^{\beta(i,k,m)} : i \in \mathcal{N}, \beta(i,k,m) \in \mathbb{N}, \text{ and } m \in \mathbb{N})$$

$$\mathcal{P}(\{e_i^{\gamma(i,k,m)} : i \in \mathcal{N}, \gamma(i,k,m) \in \mathbb{N}, \text{ and } m \in \mathbb{N})$$

where $\mathcal{P}(\cdot)$ is the power set of its argument. Then $e(k) \in \mathcal{E}$ denotes an event occurring at time k that is composed of partial events (and can therefore represent multiple simultaneous AAV passes and estimate updates).

Which events may actually occur will depend on the state of the system; specifically, a set-valued enable function $g : \mathcal{X} \to \mathcal{E}$ is defined as follows: if $e(k) \in g(x)$, then

(a) for each $e_i^{\alpha(i,k)} \in e(k)$ the following conditions hold:

(*i*) $\alpha(i,k) = 0$ if and only if $\tilde{x}_i^i(k) \leq \tilde{x}_i^{i+1}(k)$,

(*ii*) if $\tilde{x}_i^i(k) > \tilde{x}_i^{i+1}(k)$, then

$$1 \leq \alpha(i,k) \leq \left\lceil \tfrac{1}{2}(\tilde{x}_i^i(k) - \tilde{x}_i^{i+1}(k)) \right\rceil$$

(*iii*) $e_i^{\alpha'(i,k)} \notin e(k)$ if $\alpha(i,k) \neq \alpha'(i,k)$.

Condition (*i*) prevents the coordinator of node i from passing any of the AAVs on its associated surveillance area unless it thinks that its resource level is higher than the surveillance area of node $i+1$. Condition (*ii*) ensures that the coordinator of node i attempts to balance the resource level between its surveillance area and that of node $i+1$ by passing at least one AAV from the former to the latter, but not more than would make the resource level of its surveillance area less than than one unit below that of node $i+1$ after the transfer (this is more aggressive than most load balancing schemes which stipulate that a processor should not pass an amount of load that would make it less lightly loaded than its neighbors). We note that the assumption $x(k) \in \mathcal{X}_0$ and these first two conditions prevent a coordinator from passing more AAVs than actually exist on its patrol path. Condition (*iii*) simply limits the number of partial events for node i to at most one per composite event $e(k)$.

(b) for each $e_i^{\beta(i,k,m)}$ and $e_i^{\gamma(i,k,m)}$ in $e(k)$ it must be the case that $k \geq m$ to preserve causality and that both $e_i^{\beta'(i,k,m)} \notin e(k)$ if $\beta(i,k,m) \neq \beta'(i,k,m)$ and $e_i^{\gamma'(i,k,m)} \notin e(k)$ if $\gamma(i,k,m) \neq \gamma'(i,k,m)$ for the same reason as condition (*iii*) in part (a).

Now, at each time index k some event $e(k) \in g(x(k))$ occurs and alters the state of the system according to an update function $x(k+1) = f(x(k), e(k))$ defined by

$$
\begin{aligned}
x_i(k+1) &= x_i(k) - \alpha(i,k) + \alpha(i-1,k) \\
\tilde{x}_i^i(k+1) &= \tilde{x}_i^i(k) - \alpha(i,k) + \sum_{m=-\infty}^{k} \beta(i,k,m) \\
\tilde{x}_i^{i+1}(k+1) &= \tilde{x}_i^{i+1}(k) + \alpha(i,k) - \sum_{m=-\infty}^{k} \gamma(i,k,m)
\end{aligned}
$$

where the indexed α, β, and γ values here on are taken from the partial events in $e(k)$ (and we let $\alpha(i,k)$, $\beta(i,k,m)$, or $\gamma(i,k,m)$ equal zero if there is no corresponding partial event). Simply put, the update function sets node i's surveillance area's resource level at the next time index to its current level less what its coordinator passes to the next surveillance area in the graph and plus whatever is passed to it. It then updates a coordinator's estimate of resource levels according to that coordinator's actions and new information that it has received.

Let $\boldsymbol{E} \subset \mathcal{E}^{\mathbb{N}}$ be the set of all event trajectories. The set of valid event trajectories $\boldsymbol{E}_V \subset \boldsymbol{E}$ contains all event trajectories E such that there exists a valid state trajectory $X(x(0), E, k) \in \mathcal{X}^{\mathbb{N}}$ for some initial condition $x(0) \in \mathcal{X}$ (i.e. $e(k) \in g(x(k))$ for all $k \in \mathbb{N}$ and $E \in \boldsymbol{E}_V$). Whereas the enable function captures the dynamics of this system from one time step to the next, we will need to define a set of allowed trajectories in order to fully describe its behavior over time. This subset of $\boldsymbol{E}_V$, denoted $\boldsymbol{E}_B$, consists of all event trajectories that meet the following conditions:

(1) There exists $B > 0$ such that for every event trajectory $E \in \boldsymbol{E}_B$ and for any time index k, the series of events $e(k), e(k+1), \ldots, e(k+B-1)$ contains at least one occurrence of the partial event $e_i^{\alpha(i,k)}$ for all $i \in \mathcal{N}$. B is therefore the maximum number of events that may occur in the longest possible time interval between two AAV transfer decisions by a coordinator (i.e. the longest time a coordinator is permitted to ignore an resource level imbalance plus the longest time it could take for an AAV to approach the coordinator's location and be transferred). The actual value for B will depend on the number of AAVs in the system and how often they visit the coordinators.

(2) For the same constant B as above, each coordinator must detect an AAV added to or removed from the areas it joins within B time steps of when that AAV was transferred. Strictly speaking, for all $i \in \mathcal{N}$ and any $k \in \mathbb{N}$ the following must hold

$$
\sum_{n=k}^{k+B-1} \beta(i+1,n,k) = \sum_{n=k}^{k+B-1} \gamma(i-1,n,k) = \alpha(i,k),
$$
$$
\beta(i,n,k) = \gamma(i,n,k) = 0 \text{ for all } n \notin [k, k+B-1]
$$

B. Convergence Properties

For the system in Section II-A we prove that the set $\mathcal{X}_I$ is invariant and that the state of the system converges to $\mathcal{X}_I$ in finite time. We include here the important lemmas and the final convergence theorem. Proofs have been omitted due to space constraints.

Lemma 1: The set $\mathcal{X}_I$ is invariant with respect to the system in Section II-A.

Remark: Trajectories within the invariant set $\mathcal{X}_I$ are not, in general, static. When the total number of AAVs, L, is not evenly divisible by the number of nodes, N, the excess AAVs are continually passed around the ring so that the resource level of each node alternates between two consecutive integers. When L is divisible by N, however, the only state within $\mathcal{X}_I$ is the perfectly balanced state $x_i = x_j$ for all $i, j \in \mathcal{N}$.

Lemma 2: For the system described in Section II-A, the minimum resource level of the nodes is non-decreasing in time.

Lemma 3: For the system described in Section II-A the following inequalities hold

$$
\begin{aligned}
\tilde{x}_i^i(k) &\geq x_i(k-B) - \sum_{n=k-B}^{k-1} \alpha(i,n) \\
\tilde{x}_i^{i+1}(k) &\leq x_{i+1}(k-B) + \sum_{n=k-B}^{k-1} \alpha(i,n)
\end{aligned}
$$

for all $i \in \mathcal{N}$ and for all $k \geq B$, where B is the delay associated with $\boldsymbol{E}_B$.

Lemma 4: Define $m(k) \triangleq \min_i x_i(k)$ as the minimum resource level at time k and $n(k) \triangleq \left| \{i \in \mathcal{N} : x_i(k) = m(k)\} \right|$ as the number of nodes achieving that minimum resource level at time k. For the system described in Section II-A, for any time index k and any $x(k) \in \mathcal{X}_0 - \mathcal{X}_I$, there exists a finite number $T \in \mathbb{N}$ such that either $m(k+T) > m(k)$ or $n(k+T) < n(k)$. In other words, whenever the state of the system is not in the invariant set $\mathcal{X}_I$, it is eventually the case that the minimum resource level of the nodes increases or the subset of nodes with that minimum resource level decreases in size.

Theorem 1: For the AAV balancing system in Section II-A, for any initial condition $x(0) \in \mathcal{X}_0$ there exists a finite number $T_{x(0)} = 2B\lceil \frac{N}{2} \rceil (\lfloor \frac{L}{N} \rfloor + 1)$ such that $x(k) \in \mathcal{X}_I$ for all $k \geq T_{x(0)}$.

III. Simulations

We include simulations for two purposes. The first is to provide an illustrative example of the system in operation and the second is explore the effects of various parameters on the convergence time of the system through Monte Carlo trials. In all simulations, the length of time taken between an AAVs visits to the same coordinator were identical for all areas and AAVs. Passing decisions were taken by the coordinators every time they were visited by an AAV (so the coordinators were limited to either passing one AAV or none at a time).

Figure 4 shows an example scenario of the system in operation. At $t = 0$ the system is in equilibrium with each of the 10 nodes having either 10 or 11 AAVs (with a total of $L = 104$ AAVs). From $t = 0$ to $t = 5$ the non-static nature of the invariant set $\mathcal{X}_I$ is evident as the nodes fluctuate between 10 and 11 AAVs. At $t = 5$ some event occurs that results in the removal of all AAVs from nodes 1 and 2 (reducing the total number of AAVs to $L = 83$), and we can see how the system reacts so that by approximately $t = 25$ it has recovered equilibrium, albeit at a slightly lower average resource level.

The Monte Carlo simulations that were performed consisted of 50 trial runs each (a small number justified by the extremely small variance of the performance measure). The initial condition for all of these simulations involved taking an empty system and adding all of the AAVs to one node at $t = 0$ because this creates the largest initial imbalance possible and should give us a decent estimate of its worst practical performance. In Figure 5 we show the results of increasing the average resource level $\frac{L}{N}$ for systems with differing numbers of nodes. From these plots, it appears that the average convergence time of the system is approximately proportional to the logarithm of $\frac{L}{N}$ as opposed to just $\frac{L}{N}$ as predicted by the bound in Theorem 1 (which is not surprising since that theorem uses a worse case analysis). Figure 6 shows the effect on average convergence time of holding the average resource level constant and increasing the number of nodes in the system. Here the average convergence time appears to have an asymptotically linear relationship to number of nodes.

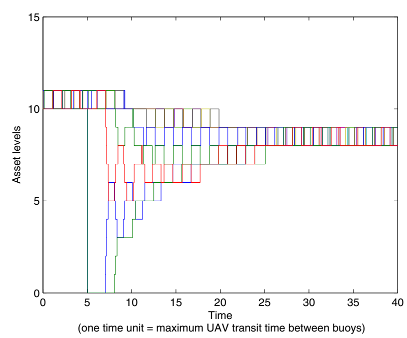

Fig. 4. resource levels in a system when a number of AAVs are removed at time $t = 5$.

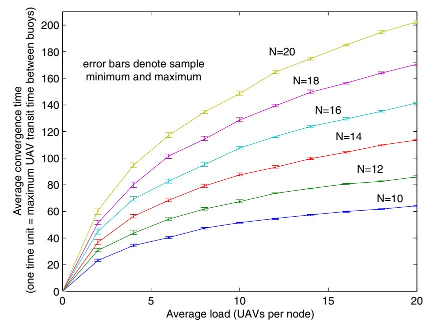

Fig. 5. Average convergence time versus average resource level for different numbers of nodes.

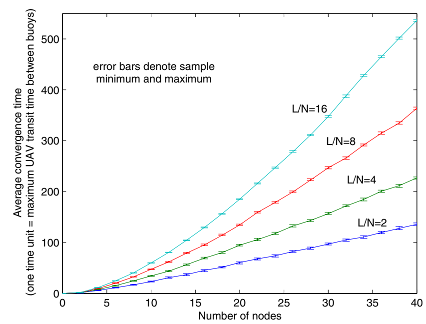

Fig. 6. Average convergence time versus number of nodes for different values of average resource level.

IV. Conclusions and Future Directions

As we have seen, the system described in Section II-A converges to a globally balanced state within a finite amount of time that can be bounded in terms of system parameters. It therefore solves the more general problem presented in Section I by restricting the allowed surveillance area interconnections to the specific topology of a unidirectional ring (and thus to any Hamiltonian topology by simple modification) Although presented in the context of an AAV mission, the algorithm developed here has a generic resource balancing application. Future directions lie in developing algorithms for area-coordinator interconnections other than a unidirectional ring, considering the case of virtual load [5], and perhaps finding a way to replace the use of coordinators with AAV to AAV communication.

References

[1] C. Schumacher, P. R. Chandler, and S. J. Rasmussen, "Task allocation for wide area search munitions via iterative network flow," in *AIAA Guidance, Navigation, and Control Conference*, no. 2002–4586, (Monterey, CA), 2002.

[2] Y. Liu, J. B. Cruz, and A. G. Sparks, "Coordinating networked uninhabited air vehicles for persistent area denial," in *43rd IEEE Conference on Decision and Control*, (Paradise Island, Bahamas), pp. 3351–3356, December 2004.

[3] E. Frazzoli and F. Bullo, "Decentralized algorithms for vehicle routing in a stochastic time-varying environment," in *43rd IEEE Conference on Decision and Control*, (Paradise Island, Bahamas), pp. 3357–3363, December 2004.

[4] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Bellmont, MA: Athena Scientific, 1997.

[5] K. L. Burgess and K. M. Passino, "Stability analysis of load balancing systems," *International Journal of Control*, vol. 61, pp. 357–393, Feb 1995.

[6] J. Finke, K. M. Passino, and A. Sparks, "Cooperative control via task load balancing for networked uninhabited autonomous vehicles," in *42nd IEEE Conference on Decision and Control*, (Maui, Hawaii), pp. 31–36, December 2003.

[7] A. Cortés, A. Ripoll, F. Cedó, M. Senar, and E. Luque, "An asynchronous and iterative load balancing algorithm for discrete load model," *Journal of Parallel and Distributed Computing*, vol. 62, pp. 1729–1746, December 2002.

[8] R. Els and B. Monien, "Load balancing of unit size tokens and expansion properties of graphs," in *SPAA '03: Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pp. 266–273, ACM Press, 2003.

[9] D. Henrich, "The liquid model load balancing method," *Journal of Parallel Algorithms and Applications, Special Issue on Algorithms for Enhanced Mesh Architectures*, vol. 8, pp. 285–307, 1996.