Abstract—This paper examines the dynamic interplay between decentralized controllers and mesh networking protocols for controlling groups of robots. A proportional controller is used to maintain robots in a formation based on estimates of the robots’ states observed through the network. The state information is propagated through the network using a flooding algorithm, which introduces topology-dependent time delays. The coupled interaction of information flow over the network with the dynamics of the robots is modeled as a linear dynamical system. With this model it is shown that systems made up of robots with stable first order dynamics are stable for all network update times, positive feedback gains, and connected communication graphs. With higher order robot dynamics it is found that stability is a complex and counter intuitive function of feedback gain and network update time. A performance metric is proposed for analyzing the convergence rate of the multi-robot system. Experiments with flying quadrotor robots verify the predictions of the model and the performance metric.

I. INTRODUCTION

In this work, we investigate the problem of controlling a group of robots over a wireless mesh network, specifically focusing on the interplay between the decentralized controller and the networking protocol. Controlling a group of robots over a wireless network involves three interacting components: decentralized control, decentralized estimation, and a networking protocol. Each of these are traditionally studied in disparate contexts using mathematical tools that are not mutually compatible. One of the chief challenges in multi-robot systems is to understand the interaction among these components under a consistent mathematical model.

Most previous studies have focused only on decentralized control, or on the coupled interaction of decentralized control and estimation, usually assuming a trivial network protocol (e.g. instantaneous communication among 1-hop neighbors). However, in experiments with groups of robots it has been observed that the dynamics of the network protocol play a critical role in the performance and stability of the desired control task. Aggressive control tasks require well-connected networks with fast update rates to maintain adequate performance and preserve stability, and if the update rate is too slow or the network too disperse, the system can become unstable. This phenomenon has been well documented, for example in [1], [2]. In this work we specifically focus on modeling a realistic networking protocol known as flooding [3], [4], coupled with a proportional feedback controller for maintaining the robots in a formation.

Using the flooding algorithm, every robot maintains an estimate of the state of every other robot in the network, which it can use in its controller. To model the separate graphical structures of the controller and the communication network we introduce a graph, distinct from the communication graph, called the control graph. Using this convenient notion we show that the multi-robot system composed of the robots, controller, and networking algorithm takes the form of a linear, discrete time dynamical system. With this model we prove two theorems concerning the stability of systems with specific robot models. We prove, surprisingly, that systems with first order robots are stable for all positive control gains, network update rates, and communication graphs. Further, we prove that networks with second order robots are stable for all positive control gains and connected communication graphs, as long as the network update time is a whole multiple of the open-loop period of the robots.

We then address the question of performance of the multi-robot system by formulating a metric to quantify the convergence rate of the system. This metric is a function of both the eigenvalues of the system model and the network update time. We validate the predictive power of the model with extensive experimental results with flying quadrotor robots. We carried out thirty two experiments with a network of three quadrotors to validate the performance metric at multiple network update rates and control gains. Details of earlier experimental results were presented in [5].

A. Related Work

Decentralized control, decentralized estimation, and mesh networking protocols have all been studied extensively in their own right, though few works have considered the effects of coupling two or more of these components together. Some recent works have studied control together with estimation. For example, in [6] the authors consider formation control with a consensus-based state estimation algorithm. The authors derive conditions for the stability of the coupled formation controller and state estimator. A related problem is considered in [7] where a group of robots estimate the state of a noise driven process using a consensus-based estimation algorithm. In [8], a Nyquist stability criterion is derived for formation control, and an estimation algorithm is proposed for finding the formation center in a distributed way.

Other works have presented important results for decentralized control using a simplified abstraction of the network protocol, usually captured as a fixed time delay, or as a
stochastic time delay. Decentralized control is considered in an $H_{\infty}$ framework in [9], where decentralized control with time delayed feedback is used as an example. In [10] and [11] results are derived for a continuous time system with discrete network updates modeled as time delays between 1-hop neighbors. In controlling platoons of vehicles in a line, [12] considers the effects of time delays between neighbors. In the study of consensus, there has been considerable work on quantifying the effects of time delays on the stability of the system, such as in [13] and [14] where maximum allowable time delays are derived. Formation control where the network protocol is abstracted as random delays is considered in [15].

A more detailed model of network induced time delays is used in [16], which looks at the coordination of ground and aerial vehicles. Also, notions of stability in leader follower formations have been studied without explicitly accounting for the network algorithm in [17]. There is also a well developed literature on controlling decentralized systems over a fixed hub-style network (as opposed to the mesh networks considered here) for which a thorough synopsis can be found in [18].

Our model differs from those above in that we consider a particular discrete time networking protocol and its coupled interaction with a continuous time controller and robot dynamics. We intentionally use a simple estimation algorithm (each robot uses the most recent state measurement available for each other robot) so as not to obscure the interacting effects of the network and the controller.

II. MODEL AND PROBLEM FORMULATION

Let there be $n$ robots, with states $x_i(t) \in \mathbb{R}^N$ that evolve in continuous time $t \in \mathbb{R}_{\geq 0}$ according to the linear time invariant (LTI) dynamics

$$\dot{x}_i(t) = A_i x_i(t) + B_i u_i(t)$$

where $u_i(t) \in \mathbb{R}^M$ is the control input. Suppose also that each robot can directly measure an output vector

$$y_i(t) = C_i x_i(t),$$

where $y_i(t) \in \mathbb{R}^P$. Furthermore, the $n$ robots communicate with one another over an ad hoc wireless mesh network.

Let the communication network be modeled in the typical way by an undirected graph $G^{\text{comm}} = (V, E^{\text{comm}})$, where each robot is a vertex in the vertex set $V$, and $E^{\text{comm}}$ is a set of pairs $(i, j)$ and $(j, i)$ for all robots $i$ and $j$ that are in communication with one another. We assume throughout the paper that $G^{\text{comm}}$ is connected with diameter $d \leq n - 1$. Each robot has a set of communication neighbors $N_i^{\text{comm}}$, which consists of the indices of the robots that share an edge with robot $i$ in $E^{\text{comm}}$. The network is used to propagate output information for each robot to each other robot. The robots then apply an estimator to these outputs to obtain a state estimate for the other robots in the network. This architecture can be used with any number of network protocols and state estimators. We choose to focus on a simple but realistic protocol known as flooding, along with a very simple estimator.

A. Network Protocol

A common way to propagate information over a wireless mesh network is to use a flooding algorithm. Flooding algorithms have many variations and can be tuned to suit specific applications, as in [4]. Here we will describe what is perhaps the simplest flooding algorithm because it can be easily modeled in a dynamical systems framework.

The main principle of the flooding algorithm is that each robot maintains an outdated copy of the outputs of all the other robots in the network. The algorithm proceeds in update steps, each of which takes time $T$. Each update step represents a cycle of a Time Division Multiple Access (TDMA) networking protocol in which each robot has a fixed time slot during which to broadcast information. When it is robot $i$’s turn to broadcast, that robot sends a packet containing its outdated outputs for all of the robots in the network, and a vector of time delays denoting when each output was current. The rest of the time, robot $i$ listens for broadcasts from other robots.

Let $\hat{y}_{ij}(t)$ be robot $i$’s outdated copy of robot $j$’s output $y_j(t')$ at some previous time $t' \leq t$. Also, let $\tau_{ij}$ denote the number of network updates that have passed since the output $\hat{y}_{ij}(t)$ was current. Denote the vector of all of robot $i$’s output estimates and time delays as

$$\hat{y}_i(t) = [\hat{y}_{i1}^T(t) \cdots \hat{y}_{in}^T(t)]^T,$$

and $\tau_i(t) = [\tau_{i1}(t) \cdots \tau_{in}(t)]^T$, respectively. Now when robot $i$ receives $\hat{y}_{ik}(t)$ and $\tau_{ik}(t)$ from robot $k$, it replaces its $\hat{y}_{ij}$ values with those of robot $k$ that are more current than its own. This can be interpreted as a very simple estimator. One might also choose, for example, to use a Kalman filter on the outputs to obtain a state estimate for every robot in the network. The flooding algorithm is shown in Algorithm 1. We emphasize that our goal is not to improve upon networking protocols, but to provide a realistic model and salient analysis of a coupled protocol and decentralized controller.

The estimate $\hat{y}_i(t)$ is constant over the interval between successive network updates $pT \leq t < (p + 1)T$ for the $p$th network update. Also, we assume that the robots know their own output without any delay, so that $\hat{y}_{ii}(t) = y_i(pT)$ for $pT \leq t < (p + 1)T$, and $\tau_{ii}(t) = 0$. Therefore $\hat{y}_{ii}(t)$ can be thought of as a zero order hold on $y_i(t)$ with sampling interval $T$. Moreover, after the algorithm has repeated a number of times equal to the diameter of the communication graph $d$, the time delay $\tau_{ij}$ associated with robot $i$’s estimate of robot $j$’s output is simply the number of hops that $j$ is from $i$ in the communication graph. We assume for the remainder of the paper that the flooding algorithm has run long enough so that this is the case (i.e. we let the flooding algorithm start at time $t = -dT$, and the robots’ motion starts at time $t = 0$). Then we have

$$\hat{y}_{ii}(t) = C_i \bar{x}_i(t) \quad \text{and} \quad \hat{y}_{ij}(t) = C_j \bar{x}_j(t - \tau_{ij}T).$$

where $\bar{x}_i(t)$ denotes the zero order hold on $x_i(t)$ with sampling interval $T$.

1In control and signal processing, a zero order hold is a transform which takes a continuous signal and returns a “stepped” version of that signal with values that are constant over intervals of a given duration.
Algorithm 1 Flooding Algorithm (implemented by robot $i$)

Require: Robot $i$ has a clock $t$ synchronized with the rest of the robots in the network.

1: initialize $\hat{y}_i = y_i(t)$ and $\tau_i = 0$
2: initialize $\hat{y}_{ij} = 0$ and $\tau_{ij} = \infty$ for $j \neq i$
3: initialize $\hat{y}_{ij}^{\text{min}} = 0$ and $\tau_{ij}^{\text{min}} = \infty$ for $j \neq i$
4: loop
5: if it is robot $i$’s turn to broadcast then
6: broadcast $\hat{y}_i$ and $\tau_i$
7: else if broadcast received from robot $k$ with $\hat{y}_k$ and $\tau_k$ then
8: for $j = 1$ to $n$ do
9: if $\tau_{kj} < \tau_{ij}^{\text{min}}$ then
10: update $\hat{y}_{ij}^{\text{min}} = \hat{y}_{kj}$ and $\tau_{ij}^{\text{min}} = \tau_{kj}$
11: end if
12: end for
13: end if
14: if TDMA round has finished then
15: update $\hat{y}_{ij} = \hat{y}_{ij}(t)$ and $\tau_{ij}^{\text{min}} + 1$ for $j \neq i$
16: update $\hat{y}_{ij} = \hat{y}_{ij}$ and $\tau_{ij} = \tau_{ij}^{\text{min}}$ for $j \neq i$
17: end if
18: end loop

The flooding algorithm provides a means by which any robot can use any other robot’s output information (albeit outdated) in its controller. This allows for the distinction between a communication graph $\mathcal{G}^{\text{comm}}$ and a control graph $\mathcal{G}^{\text{cont}}$ as described in the following section.

B. Communication Graphs and Control Graphs

Control laws for our multi-robot system can be seen as having a graphical structure separate from the communication graph. We define the control graph $\mathcal{G}^{\text{cont}} = (\mathcal{V}, \mathcal{E}^{\text{cont}})$ to be a directed graph in which each robot is identified with a node, and if the delayed output of a robot $j$ is used in the computation of the control law of robot $i$, then there is a directed edge $(j, i) \in \mathcal{E}^{\text{cont}}$.

When considering the control graph it becomes apparent that a particular control task might be implemented over a number of different control graphs, for the same communication graph. For example, if we want robots to maintain the formation of a single file line, we might have a controller that controls each robot to remain a distance $\delta$ from the neighbor immediately in front of it, or we might control the $i$th robot to remain a distance $\delta(i-1)$ from the first robot in the line. The first controller induces a chain shaped control graph, as in the top of Fig. 1, and the second induces a star shaped control graph, as in the bottom of Fig. 1, however both have the same ultimate objective. This provokes the question: what is the best control graph to accomplish a given control task over a given communication graph? We point towards a solution to this question by proposing a performance metric in Sec. IV.

C. Coupled Network and Control System

In modeling the closed loop controller and network algorithm, we must account for the fact that when robot $i$ computes its controller, it is using an output value for $j$ that has been routed over the communication network, and that this route will have taken a certain number of update times $T$. It has been observed in many practical multi-robot control scenarios that this time delay is a significant factor, and can lead to instability under some communication and control graph topologies and network update times.

Consider a formation controller of the form

$$u_i = \sum_{j \in N^\text{cont}_{i}} K_{ij}(\hat{y}_{ij}(t) - y_i(t) - \delta_{ij}), \quad (3)$$

where $\delta_{ij}$ defines the desired relative outputs $y_j - y_i$ for the formation. We consider only proportional output feedback in this paper, though similar results can be obtained for dynamic controllers as in [8]. Further divide the index set into subsets $N^\text{cont}_{i\tau}$ of all control neighbors $j$ that are $\tau$ hops away from $i$ in the communication graph, so that $N^\text{cont}_{i\tau} = \cup_{\tau=1}^{\tau} N^\text{cont}_{i\tau}$ and $N^\text{cont}_{i\tau} \cap N^\text{cont}_{i\tau'} = \emptyset$ for all $\tau \neq \tau'$. Some $N^\text{cont}_{i\tau}$ may be empty if there are no control neighbors $\tau$ hops away from $i$.

This construction is illustrated graphically in Fig. 2.

The control input for each robot can be divided into a continuous component $K_{ii}\hat{y}_i(t)$, where $K_{ii} = \sum_{j \in N^\text{cont}_{i}} K_{ij}$ and a component that is constant over the time step of the network $\sum_{\tau=1}^{\tau} \sum_{j \in N^\text{cont}_{i\tau}} K_{ij}(\hat{y}_{ij}(t) - \delta_{ij})$. Recall from Section II-A that the delay of $\hat{y}_{ij}(t)$ is equal to $T$ times the number of hops that $j$ is away from $i$ in the communication graph. We can rewrite the system using (2) as

$$\hat{x}_i(t) = (A_i - B_iK_{ii}C_i)x_i(t) + B_i\sum_{\tau=1}^{\tau} \sum_{j \in N^\text{cont}_{i\tau}} K_{ij}(C_j\bar{x}_j(t - \tau T) - \delta_{ij}), \quad (4)$$

Fig. 1. Elementary example of two different control graphs that can be used to carry out the same control objective.

Fig. 2. This figure illustrates how the control neighborhood of a robot $N^\text{cont}_{i\tau}$ can be divided into sub-neighborhoods $N^\text{cont}_{i\tau'}$ of all robots that are $\tau$ hops away from $i$ in the communication graph.
Now the system looks like a continuous time system with a constant input over intervals of length $T$. We can write the continuous time dynamics as a discrete time equation by integrating over the intervals of length $T$. This gives the value of the continuous state $x_i(t)$ at $T$ second intervals as

$$x_i(t + T) = \Phi_t x_i(t) + \Gamma_i B_i \sum_{\tau = 1}^{d} \sum_{j \in N_i^{\text{cont}}} K_{ij} (C_j x_j(t - \tau T) - \delta_{ij}).$$

(5)

where

$$\Phi_t = e^{(A_i - B_i K_i C_i) T}$$

and

$$\Gamma_i = \int_0^T e^{(A_i - B_i K_i C_i) s} ds,$$

(6)

and the exponentials are matrix exponentials [19]. We emphasize that this is not an approximation; it is the analytically determined value of the state at discrete time instants.

Now define a state vector $x(t) = [x^T_1(t) \cdots x^T_n(t)]^T$, which is the concatenation of the states $x_i(t)$, and $X(t) = [x^T(t) \cdots x^T(t - dT)]^T$, which is the concatenation of all the time delayed states up to the maximum number of hops it can take for a packet to cross the network, namely $d$. The vector $X$ is the state of the coupled network protocol and robots. Note that it has $N \times n \times (d + 1)$ elements, which is $(d + 1)$ times the number of elements there would be were we to ignore the network protocol. Because the robot dynamics, controller, and network algorithm are linear, the entire coupled system is linear and can be written

$$X(t + T) = AX(t) + \Delta,$$

(7)

where

$$A = \begin{bmatrix} \Phi & \Psi_1 & \cdots & \Psi_d \\ I_{Nn} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & I_{Nn} & 0 \end{bmatrix},$$

(8)

with $\Phi = \text{diag}([\Phi_1 \cdots \Phi_n])$, $\Psi_\tau = [\kappa_{ij}^{\tau}]$, and

$$\kappa_{ij}^{\tau} = \begin{cases} \Gamma_i B_i K_{ij} C_j & \text{if } j \in N_i^{\text{cont}} \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\Delta = \begin{bmatrix} - (\Gamma_1 B_1 \sum_{j \in N_1^{\text{cont}}} K_{ij} \delta_{ij})^T \\ \vdots \\ - (\Gamma_n B_n \sum_{j \in N_n^{\text{cont}}} K_{ij} \delta_{ij})^T 0 \cdots 0 \end{bmatrix}^T,$$

(9)

where $I_{Nn}$ is the $Nn \times Nn$ identity matrix, and 0 is a matrix of zeros of the appropriate dimensions.

Our goal in this work is to determine useful conditions on the network update time $T$, the feedback control gains $K_{ij}$, and the control graph $G^{\text{cont}}$ to guarantee the stability and convergence of the system in (7) for a given robot dynamics and a given communication graph.

### III. Stability Conditions

In this section we present the main analytical results of the paper. Using the model from Section II we derive a stability condition for the general case, and then find more targeted results for the case of first order and second order robot dynamics.

#### A. General Stability Condition

The condition for the stability (in the sense of Lyapunov) of the closed loop system is that all of the eigenvalues of $A$ lie within the closed unit circle, that is $|\lambda_i(A)| \leq 1$ for all $i$. For asymptotic stability, the condition is $|\lambda_i(A)| < 1$ for all $i$. The eigenvalues of $A$ are functions of $T$ and $K_{ij}$, and the structure of $A$ is dictated by the control and communication graphs (i.e. the $N_e^{\text{comm}}$ neighborhoods determine which entries in $\Psi_\tau$ are non-zero). We can simplify the problem of determining the eigenvalues of $A$ considerably with the following theorem.

**Theorem I (System Eigenvalues):** The $Nn(d + 1)$ eigenvalues of the multi-robot system transition matrix $A$ in (8) are given by the solutions to the equation

$$\det(M^{d+1} I_{Nn} - \left(\lambda^d \Phi + \sum_{\tau=1}^{d} \lambda^{-\tau} \Psi_\tau\right)) = 0.$$

(9)

**Proof:** The eigenvalues of $A$ are given by the solutions of the characteristic equation $\det(M^{d+1} I_{Nn(d+1)} - A) = 0$. We will simplify this by exploiting the block structure of $A$, and using the identity $\det(M) = \det(M_{22}) \det(M_{11} - M_{12} M_{22}^{-1} M_{21})$, where

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}.$$  

Define $M = \lambda I_{Nn(d+1)} - A$ and notice that it has the desired block structure with $M_{11} = \lambda I_{Nn} - \Phi$, $M_{12} = -[\Psi_1 \cdots \Psi_d]$, $M_{21} = -[I_{Nn} 0 \cdots 0]^T$, and

$$M_{22} = \lambda I_{Nnd} - \begin{bmatrix} 0 & 0 \\ I_{Nn(d-1)} & 0 \end{bmatrix}.$$  

The block $M_{22}$ is lower triangular, so its determinant is the product of the diagonal entries, namely $\det(M_{22}) = \lambda^{Nnd}$. The term $M_{12} M_{22}^{-1} M_{21}$ can be computed to be

$$M_{12} M_{22}^{-1} M_{21} = \sum_{\tau=1}^{d} \lambda^{-\tau} \Psi_\tau,$$

and applying the identity gives the considerably simpler characteristic equation

$$\lambda^{Nnd} \det \left(\lambda I_{Nn} - \Phi - \sum_{\tau=1}^{d} \lambda^{-\tau} \Psi_\tau\right) = 0,$$

and bringing the $\lambda^{Nnd}$ inside the determinant gives the desired result.

For stability, the solutions of (9) must lie within the closed unit circle. One can check this condition straightforwardly by computational means for particular systems and controllers. Unfortunately, the condition is too general to be of use as a rule of thumb, or to give an insight into the trade-offs between control gain and network update rate. For this, we seek special cases that are of particular interest.
B. Relation to the Control and Communication Graphs

It is typical in control problems with a graphical structure to have algebraic properties of the graph, such as the graph Laplacian or the adjacency matrix, appear in the system equations. One may rightly wonder where these graph properties are in our model, especially because our system involves two graphs, the control graph and the communication graph.

The adjacency matrix $\Lambda(G)$ of a directed graph $G$ is the matrix which has a 1 for the entry $(i,j)$ if there is a directed edge from $j$ to $i$ in the graph (that is, $j$’s output is used in $i$’s controller), and a zero otherwise. Let the adjacency matrix of the control graph be given by $\Lambda^{\text{cont}} = \Lambda(G^{\text{cont}})$. Now define a sequence of adjacency matrices $\Lambda^{\text{comm}}_\tau$ which are the adjacency matrices of the graph of $\tau$-hop neighbors in the communication graph. That is, there is a 1 in element $(i,j)$ of $\Lambda^{\text{comm}}_\tau$ if the shortest path between $j$ and $i$ in the communication graph is $\tau$ hops long. We can re-write the equation (9) in terms of these adjacency matrices.

Let $\circ$ denote the element-wise product of two matrices, and $\otimes$ the Kronecker product. Then $\lambda^{d-\tau} \Lambda^{\text{comm}} \circ \Lambda^{\text{cont}}$ is a matrix with $\lambda^{d-\tau}$ in any entry $(i,j)$ which is in the control adjacency matrix and is $\tau$ hops away in the communication graph. Furthermore, if we assume that all robots have the same dynamics and the same number of control neighbors, so $\Gamma_i = \Gamma$, $B_j = B$, and $C_i = C$ for all $i$, and that all robots use the same feedback gain, so that $K_{ij} = K$ for all $(i,j)$ in the control graph, then we have $\lambda^{d-\tau} \Psi = \lambda^{d-\tau} \Lambda^{\text{comm}} \circ \Lambda^{\text{cont}} \otimes \Gamma BKC$.

Although this notation shows more clearly the relation of the characteristic equation in (9) to the structure of the communication and control graphs, we will continue to use the previous notation because it is more general and more compact.

C. First Order Robot Model

Consider the case of $n$ one dimensional robots modeled as stable first order systems

$$\dot{x}_i = -a_i x_i + b_i u_i, \quad y_i = c x_i,$$

where $a_i \geq 0$, $b_i > 0$, with controllers

$$u_i = \sum_{j \in \mathcal{N}^{\text{cont}}_i} k_{ij} (y_{ij}(t) - y_i(t) - \delta_{ij}),$$

for scalar feedback gains $k_{ij} > 0$. We require that the scalar $c$ be the same for all robots. This class of robot models, especially the simple sub-case of integrator robots (in which the robot’s velocity is controlled directly), is common in the multi-robot control literature. For this model, the closed loop single robot discrete time dynamics from (5) take the form

$$x_i(t + T) = \phi_i x_i(t) + \frac{(1 - \phi_i)}{(a_i + b_i k_{ii} c)} \sum_{\tau = 1}^{d} \sum_{j \in \mathcal{N}^{\text{cont}}_i} b_j k_{ij} c (x_j(t - \tau T) - \delta_{ij}),$$

where $\phi_i = e^{-(a_i + b_i k_{ii} c)T}$. Notice that $0 < \phi_i < 1$ for $a_i \geq 0$, $b_i k_{ii} c > 0$, and $T > 0$.

Now the state transition matrix $A$ for the closed loop robot network can be written from (8) as

$$A = \begin{bmatrix}
\Phi & \Psi_1 & \cdots & \Psi_d \\
I_n & 0 & \cdots & 0 \\
\vdots & \ddots & \ddots & \vdots \\
0 & \cdots & I_n & 0
\end{bmatrix},$$

where $\Phi = \text{diag}([\phi_1 \cdots \phi_n])$, and $\Psi_\tau = [\kappa_{ij}^\tau]$ with

$$\kappa_{ij}^\tau = \begin{cases}
(1-\phi_i) b_j k_{ij} c, & \text{if } j \in \mathcal{N}^{\text{cont}}_i \\
0, & \text{otherwise.}
\end{cases}$$

To give the reader an idea of the size of this system, depending on the diameter of the communication graph $d$, the state is at least $2n$ dimensions and at most $n^2$ dimensions.

Surprisingly, the network of first order robots is stable for any positive network update time $T$, any positive control gains $k_{ij}$, and any control graph, as formalized in the following theorem.

**Theorem 2 (First Order Robot Networks are Stable):**
For first order robots (10) with controllers (11) using the flooding algorithm (Algorithm 1) to update $y_{ij}$ over a static, connected communication network, the multi-robot system is Lyapunov stable for all $k_{ij} > 0$, $T > 0$, and for all control graphs.

**Proof:** The proof follows from Geršgorin’s circle theorem, which states that all the eigenvalues of an $m \times m$ matrix lie within the union of the $m$ closed circles on the complex plane centered at the diagonal elements with radii equal to the sum of the absolute value of the off diagonal elements in the row. Specifically, given an $m \times m$ matrix $A = [a_{ij}]$, for all eigenvalues $\lambda_i(A)$ there exists $j \in \{1, \ldots, m\}$ such that $|\lambda_i(A) - a_{jj}| \leq \sum_{k=1, k \neq j}^{m} |a_{jk}|$, where $|\cdot|$ is the magnitude of a complex number.

In our case, the first $n$ diagonal elements of $A$ are $\phi_i$, and the absolute sum of the off diagonal elements of the first $n$ rows are

$$0 \leq \frac{(1 - \phi_i) b_j k_{ij} c}{(a_i + b_i k_{ii} c)} \leq (1 - \phi_i),$$

using the fact that $a_i \geq 0$ and $b_i k_{ii} c > 0$. This corresponds to $n$ circles centered at $\phi_i$ extending in the positive real direction to a point less than or equal to 1, since $\phi_i + \frac{(1-\phi_i) b_j k_{ij} c}{(a_i + b_k k_{ii} c)} \leq \phi_i + (1 - \phi_i) = 1$. They extend in the negative real direction to a point greater than or equal to -1, since $\phi_i - \frac{(1-\phi_i) b_j k_{ij} c}{(a_i + b_i k_{ii} c)} \geq \phi_i - (1 - \phi_i) > -1$. We conclude that these $n$ circles are contained in the closed unit circle.

The last $(n-1)$ diagonal elements are 0, and the off diagonal absolute sums are 1, therefore these circles are also contained in the unit circle (they are equal to it). We therefore
conclude that all $n^2$ eigenvalues lie on or within the unit circle regardless of control gain, network update time, or control graph topology.

Integrator robots are a particularly common model in the multi-robot literature, therefore we state the following corollary.

**Corollary 1 (Integrator Robot Networks are Stable):**
For integrator robots $\dot{x}_i(t) = u_i(t)$, with controllers $u_i = \sum_{j \in \mathcal{N}^\text{cont}} k_{ij} (\dot{x}_j(t) - x_i(t) - \delta_{ij})$ and using Algorithm 1 to update $\dot{x}_i(t)$ over a static, connected communication network, the multi-robot system is Lyapunov stable for all $k_{ij} > 0$, $T > 0$, and for all control graphs.

*Proof:* Apply Theorem 2 to the case with $a_i = 0$, $b_i = 1$, and $c = 1$.

**Remark 1 (Extension to N-Dimensional Robots):**
The extension of Theorem 2 to $N$-dimensional robots with $A_i$, $B_i$, and $K_{ij}$ diagonal is immediate. When $A_i$, $B_i$, and $K_{ij}$ are not diagonal the situation is considerably more complex.

**Remark 2 (Significance and Interpretation):** Theorem 2 indicates that integrator and first-order robot models, which are common in the multi-robot control literature, are not rich enough to exhibit instability due to network time delays. The main factor lying behind this result is the composition of continuous time robot dynamics with discrete-time network dynamics. The continuous to discrete transform in (12) results in coefficients $\phi_i$ on the state and $(1 - \phi_i)$ on the input, which balance each other in such a way that instability is impossible.

**D. Second Order Robot Model**

To investigate the stability properties of networks with more realistic robot dynamics, we turn to the case of a second-order robot model. Consider a one degree of freedom second-order robot with position $p_i(t)$ and state $x_i(t) = [p_i(t) \ p_i(t)^T]$. Let the dynamics of the robot be given by

$$A_i = \begin{bmatrix} 0 & 1 \\ -c & b \end{bmatrix}, \quad B_i = \begin{bmatrix} 0 \\ c \end{bmatrix}, \quad C_i = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

and $K_{ij} = k$, with $b$, $c$, and $k$ positive scalars. This is a fair model of the translational dynamics of a flying quadrotor robot that is stabilized about its angular axes with an onboard controller, for example [5], [20].

For this system, we can find the matrices $\Phi_i$ and $\Psi_i$ analytically by solving the second-order response over the $T$ time window. Suppose for now that $k > (b^2 - c) / (4|\mathcal{N}^\text{cont}|c)$. This will ensure that the system response is oscillatory. The response of the position as a function of the update time $T$ can be found by solving the second-order ODE to get

$$p_i(t + T) = e^{-\sigma T} \left( p_i(t) \cos(\omega_i T) + \frac{\sigma p_i(t)}{\omega_i} \sin(\omega_i T) \right) + \left( \frac{\sigma^2 p_i(t)}{\omega_i} + \frac{p_i(t)}{\omega_i} - \frac{\sigma p_i(t)}{\omega_i} \right) \sin(\omega_i T),$$

where $\sigma = b/2$ and $\omega_i = \sqrt{\sigma^2|\mathcal{N}^\text{cont}|k + 1 - (b/2)^2}$. Differentiating with respect to $T$ yields

$$p_i(t + T) = e^{-\sigma T} \left( p_i(t) \cos(\omega_i T) + \frac{\sigma^2 p_i(t)}{\omega_i} + \frac{p_i(t)}{\omega_i} - \sigma p_i(t) \right) \sin(\omega_i T),$$

from which the transition matrix can be found to be

$$\Phi_i = e^{-\sigma T} \times \begin{bmatrix} \cos(\omega_i T) + \frac{\sigma}{\omega_i} \sin(\omega_i T) & \frac{1}{\omega_i} \sin(\omega_i T) \\ \left( \frac{\sigma^2}{\omega_i} + \omega_i \right) \sin(\omega_i T) & \cos(\omega_i T) - \frac{\sigma}{\omega_i} \sin(\omega_i T) \end{bmatrix}.$$ (14)

The matrix $\Gamma_i$ can be found from the relation $I + A_i \Gamma_i = \Phi_i$, (which follows from the definitions of $\Phi_i$ and $\Gamma_i$ in (6)). In our case, the matrix $A$ is invertible, so we can find $\Gamma_i$ from $\Gamma_i = A^{-1}(\Phi_i - I)$, and for $j \in \mathcal{N}_i$ we can compute

$$\kappa_{ij}^T = \Gamma_i B_i k C_j$$

**Remark 3 (Second Order Stability):**
For robots with second-order dynamics given by (13), using the controller in (3), and using Algorithm 1 to update $\dot{y}_{ij}(t)$ over a static, connected communication network, the following conditions are sufficient for the stability of the multi-robot system:

1. All nodes in the control graph have in-degree $|\mathcal{N}^\text{cont}_i|$, and for all $i$, $k > \frac{b^2 - c}{4|\mathcal{N}^\text{cont}_i|c}$.
2. $T = \frac{2\pi m}{\omega_i}$ for some positive integer $m$.

*Proof:* We again prove the result using Geršgorin’s circles. The first condition ensures that $|\mathcal{N}^\text{cont}_i| = |\mathcal{N}^\text{cont}_i|$ for all $i$, which implies that the transition matrix $\Phi_i$ is the same for all robots, and that $\kappa_{ij}^T$ is the same for all $i$ and for all $j \in \mathcal{N}_i$. The second condition ensures that the system response for all robots is oscillatory, with transition matrix given by (14) and with $\kappa_{ij}^T$ given by (15).

The third condition says that $T = 2\pi m / \omega_i$. Substituting this into (14) yields $\Phi_i = I_2 e^{-\sigma T}$, and in (15) it gives

$$\kappa_{ij}^T = \begin{bmatrix} k & 0 \\ 0 & 0 \end{bmatrix},$$

if $j \in \mathcal{N}_i$ and $\kappa_{ij}^T = 0$ otherwise. Therefore, the first 2n rows of $A$ have 1 on the diagonal and have off diagonal absolute row sums of

$$\frac{|\mathcal{N}^\text{cont}_i| k}{|\mathcal{N}^\text{cont}_i| k + 1} (1 - e^{-\sigma T}),$$

for the odd rows and 0 for the even rows. Thus the circles corresponding to the first $n$ odd rows are centered at $e^{-\sigma T}$ and they extend to $e^{-\sigma T} + \frac{|\mathcal{N}^\text{cont}_i| k}{|\mathcal{N}^\text{cont}_i| k + 1} (1 - e^{-\sigma T}) < e^{-\sigma T} + (1 - e^{-\sigma T}) = 1$ in the positive real direction and $e^{-\sigma T} - \frac{|\mathcal{N}^\text{cont}_i| k}{|\mathcal{N}^\text{cont}_i| k + 1} (1 - e^{-\sigma T}) > e^{-\sigma T} - (1 - e^{-\sigma T}) > -1$ in the negative real direction, so they are contained in the closed unit circle. The circles corresponding to the first $n$ even rows have radius zero and are centered at $0 < e^{-\sigma T} < 1$, and
the remaining \(2(n - 1)\) circles are unit circles. Therefore all eigenvalues of \(A\) are contained in the closed unit circle.

**Remark 3 (Significance and Interpretation):** The theorem exploits the fact that, when sampled at their natural period, the robots look like first order systems, so we can apply the same kind of analysis as in the first order case. The assumptions that all robots have identical dynamics and that the control graph is regular ensure that the system has only one natural period. Also, the proof does not hold if the network is updated at multiples of half the period, since then the first \(n\) odd Geršgorin’s circles extend beyond \(-1\) on the real axis. Also, Geršgorin’s circles are known to be conservative; there are many cases that do not meet the conditions of the proof that will still be stable.

**IV. PERFORMANCE METRIC**

Until this point we have been focused primarily on modeling and stability of the group of robots. Now we turn to the question of performance. If we wish to drive a group of robots in an aggressive manner, this will require fast dynamics for the coupled robots and communication system. Typically, for a stable discrete time system of the form \(x(t + 1) = Ax(t)\), the speed of response of the system is captured by the modulus of the largest eigenvalue \(|\lambda_{\text{max}}(A)|\). The smaller this value is, the faster the transient response of the system. In particular, we can use the bound \(|x(t + 1)| \leq |\lambda_{\text{max}}(A)||x(t)||\), leading to the conclusion that \(|x(t)| \leq |\lambda_{\text{max}}(A)||x(0)||\), for the initial condition \(x(0)\). That is, the size of the state is bounded by a geometric series with convergence rate \(|\lambda_{\text{max}}(A)|\).

In our case, however, the sequence is governed by the law \(X(t + T) = AX(t) + D\). As a technicality, we must change this to a homogeneous equation of the form \(\tilde{X}(t + T) = A\tilde{X}(t)\) by employing the change of variables \(X = X - X_{ss}\), where \(X_{ss}\) is the steady state value of \(X(t)\), which satisfies the equation \((I - A)X_{ss} = D\). Now the trajectory of the system can be bounded as \(|\tilde{X}(t + T)| \leq |\lambda_{\text{max}}(A)||\tilde{X}(t)||\), which leads to

\[
|\tilde{X}(t)| \leq |\lambda_{\text{max}}(A)|^{t/T}||\tilde{X}(0)||,
\]

where \(t = 0, T, 2T, \ldots\). Therefore, in our case the convergence rate of the system, which is the primary indicator of speed of response, is \(|\lambda_{\text{max}}(A)|^{1/T}\). For this reason we propose the following performance metric for comparing or optimizing the speed of networked multi-robot systems.

**Definition 1 (Convergence Rate):** The convergence rate of the multi-robot system \(X(t + T) = AX(t) + \Delta\) is given by \(\alpha = |\lambda_{\text{max}}(A)|^{1/T}\).

Notice that a necessary and sufficient condition for Lyapunov stability of the system is that \(\alpha \leq 1\), and for asymptotic stability the condition is \(\alpha < 1\). We can use this metric as a design criterion, by requiring that \(\alpha \leq \alpha^*\) given some required convergence rate \(\alpha^*\).

Example plots of the convergence rate as a function of the network update time \(T\) are shown in Fig. 3 for three quadrotor robots. For sufficiently low control gain \((K_{ij} = 3\) in this case), and for \(T\) small (compared to the natural frequency of the robots), the convergence rate agrees with the intuition that faster network updates give better performance. For higher gains \((K_{ij} = 4\) in this case) this intuition breaks down, as a slower network update time can actually improve performance. For example, in Fig. 3 for \(K_{ij} = 4\), increasing the update time from \(T = 0.4s\) to \(T = 0.6s\) significantly improves the system performance (indeed, it makes the system stable). These performance predictions were verified experimentally, as described in the next section.

**V. QUADROTOR EXPERIMENTS**

We carried out 32 experimental trials with three quadrotor robots in a Vicon motion capture environment. A fourth order model was used to represent the quadrotors with the matrices \(A_i\) and \(B_i\) derived from humans. The controller used only the relative positions of neighbors, not their full state. The experiments were performed on a quadrotor experiment protocol prior to the experiments. The controller used only the relative positions of neighbors, not their full state. The convergence rates for all 32 trials are shown in Fig. 3, along with an inset diagram showing the control (dotted) and communication (solid) topologies.

![Fig. 3. The convergence rate \(\alpha = |\lambda_{\text{max}}(A)|^{1/T}\) as a function of network update time \(T\) is shown for two gain values \(K_{ij}\) for a network of three quadrotor robots. The solid curves show the convergence rates predicted by our model, while the dots indicate experimentally determined rates, and the red circles correspond to the experiments shown in Fig. 4. The inset graph diagram shows the control (dotted) and communication (solid) topologies.](image)

**VI. CONCLUSIONS**

In this paper we derived a new model for networked multi-robot systems, placing special attention on the dynamic interaction between the networking protocol and the decentralized controller. The resulting model is a discrete time
linear dynamical system whose eigenvalues are functions of the network update time, the control gain, and the control and communication graphs of the system. We derive sufficient conditions for stability for the case of first order and second order robot dynamics. We also propose a performance metric to quantify the convergence rate of the system. The predictions of the model were verified in 32 experiments with three quadrotor robots. In the future we would like to derive stronger conditions for the stability of the multi-robot system, and to carry out trajectory following experiments.

REFERENCES


