2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Decentralized Grid-based Algorithms for Formation Reconfiguration and Synchronization

Damjan Miklic, Stjepan Bogdan, and Rafael Fierro

Abstract— **In this paper we present a formation synchronization and reconfiguration scheme based on distributed computation. In our previous work, we have introduced a formation abstraction by means of a virtual rectangular grid. The grid was used to ensure collision-free transitions between formations, coordinated by a centralized controller. This paper is an extension of that work, proposing control laws that are computed in a distributed way while still ensuring collision-free formation reconfigurations. Furthermore, we consider an extension of the group rendezvous problem. The agents are required to meet at the rendezvous point and establish consensus on formation state. Again, the grid abstraction is used to describe the formation state. Nonholonomic constraints of agent motion are taken into account explicitly. As our approach is communication-based, we also examine the effects of temporary communication loss.**

I. INTRODUCTION

In the vast majority of possible application areas of autonomous systems, a group of units working together in a coordinated fashion will by far outperform a single unit working on the same task. Examples include cooperative transportation, exploration and mapping, environment monitoring, and supporting search and rescue operations, to name just a few. Due to rapid developments in the areas of embedded computing, drive systems, sensing and wireless communications, using a group of autonomous agents to solve real-world tasks is quickly becoming technologically and economically viable.

Deploying a team of autonomous agents poses significant challenges in terms of coordination, communication and control. Those challenges have been in the focus of attention of the research community for over a decade, [1] - [2]. The fundamental issues of system stability, convergence to the desired state and robustness must be considered. High system dimensionality, complex interactions, inherent parallelism and uncertainties make analysis and control synthesis a challenging task. One often used approach is to introduce artificial potential functions [3] attached to agents and objects in the environment. These approaches have analytically provable stability, and can also ensure collision avoidance between the agents themselves, as well as between the agents and the environment. However, the potential functions also impose artificial constraints on the system and suffer from the local minima problem. Graph theory is another tool that provides a very natural description of group configuration [4] and agent interactions. Authors in [5] and [2] propose an abstraction map to transform the high-dimensional state space into a lower-dimensional space capturing the position, orientation and shape of the team. Optimization-based approaches [6] have also been successfully applied to multi-agent systems. Problems related specifically to communication, coordination and consensus have been receiving a considerable amount of attention recently [7], [8].

In spite of the significant progress that has been achieved, some important issues still remain unresolved. Most works assume a linear, first or second order model of the system, and only few (notably [2]) take into account nonholonomic vehicle constraints. Few approaches, with the exception of the potential-field based ones, can guarantee collision avoidance within the group, or with the environment. Consensus seeking and area coverage approaches [8] can rarely account for motion and orientation of the group.

Our previous work in [9], [10] introduces a formation coordination framework based on a discretized, rectangular grid-based representation of the group. Using discrete event scheduling techniques that have been well-established in manufacturing systems control, we have designed a controller that guarantees collision-free transitions between arbitrary formations. This controller was integrated with continuous-time vehicle position controllers into a hybrid control architecture, capable of driving the group through the environment while switching between formations. We now extend that approach to a more decentralized architecture. In this new approach, each agent performs the computations required to control its motion and negotiates for priority with other agents when moving through the formation. In this way, inter-agent collisions are still explicitly prevented. The only remaining centralized aspect of the proposed control system is a higher-level supervisor that provides group waypoints and formation setpoints. Furthermore, in this work we also take into account the nonholonomic constraints of vehicle motion.

The paper is organized as follows. In Section II we formally state the two problems we are considering and outline the assumptions that we are making on the agents and on the environment. Section III deals with the rendezvous and group synchronization problem. We propose decentralized

The work of S. Bogdan and D. Miklic is supported by the Ministry of Science, Education and Sports of the Republic of Croatia, grant #036-0363078-3016 ”Task Planning & Scheduling in Robotic and Autonomous Systems”.

The work of R. Fierro is supported by NSF grants ECCS CAREER #0811347, IIS #0812338, CNS #0709329, and by DOE University Research Program in Robotics (URPR).

S. Bogdan and D. Miklic are with Faculty of Electrical Engineering and Computing, University of Zagreb, 10000 Zagreb, Croatia `damjan.miklic,stjepan.bogdan@fer.hr`

R. Fierro is with the Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131, USA `rfierro@ece.unm.edu`

algorithms for formation reconfiguration in Section IV and in Section V we present simulation results showing the validity of our approach. Concluding remarks and directions for further research are given in Section VI.

II. PROBLEM STATEMENT

We are aiming to design control laws that will enable a group of autonomous agents to establish and maintain a desired formation and switch between formations in a safe and reliable manner. This should be performed in a distributed fashion, meaning that each agent carries out the computations necessary to control its own motion. To achieve this goal, we must design a controller that enables the agents to establish and maintain a common representation of group state. Furthermore, we need algorithms for switching between arbitrary formations, which are defined with respect to the common group representation. We now proceed to define the agents and environment that we are dealing with and give a more formal description of the above problems.

Let $\mathcal{A} = \{a_n | n = 1, \ldots, N\}$ be a set of N identical autonomous agents, moving in the two-dimensional plane. Each agent has the state $\mathbf{a}_n = [x_n \; y_n \; \theta_n]^T \in SE(2)$, describing its position and orientation with respect to a fixed world frame W. Agent dynamics are described by the unicycle model

$$\begin{aligned} \dot{x}_n &= v_n \cos\theta_n \\ \dot{y}_n &= v_n \sin\theta_n \\ \dot{\theta}_n &= \omega_n \end{aligned} \tag{1}$$

where v_n and ω_n are the linear and angular velocities respectively, representing control inputs of agent a_n. All agents are assumed to be traveling within a bounded region $\mathcal{S}$ with a boundary $\partial\mathcal{S}$. This region contains a finite number of static obstacles which compose the set $\mathcal{O}$, and the union of all points on the perimeter of any obstacle form the set $\partial\mathcal{O}$. Furthermore, there is a set of P predefined *waypoints* $\mathcal{Q} = \left\{\mathbf{q}_p | \mathbf{q}_p = [x_{qp} \; y_{qp}]^T \in \mathcal{S} \cap \overline{\mathcal{O}}, p = 1, \ldots, P\right\}$ that the agents must pass through. The first waypoint, $\mathbf{q}_1$ has a special purpose and is called the *rendezvous point*. We require that the space between initial agent positions and the rendezvous point be obstacle-free.

To describe the desired formation, we introduce the notion of a virtual rectangular grid $\mathcal{G}_n$ attached to each agent. We call this grid the *formation grid*. Each grid consists of $(2N+1) \times (2N+1)$ square cells, each with side length l_c. The size of the grid can be chosen arbitrarily, as long as the total number of cells is equal to or exceeds the number of agents N. Simulations have shown that $(2N+1)^2$ cells give adequate freedom in specifying formation layouts. The minimum cell size must be chosen so that the entire physical dimensions of an agent can fit within one cell. Formation layouts on the grid are represented as binary matrices $\mathbf{F} = \{f_{ij} | i, j = 1, \ldots, (2N+1)\}$, where $f_{ij} = 1$ denotes an occupied cell. Conversely, the position of each agent on the grid is described with a pair of *grid coordinates* $\mathbf{a}_n^{(G)} = [i_n \; j_n]^T$, $i_n, j_n \in \{1, \ldots, 2N+1\}$, relative to the upper left corner of the grid[1]. Grid position and orientation in $\mathcal{S}$ are given by the $[x_{gn} \; y_{gn} \; \theta_{gn}] \in SE(2)$ triplet.

Taking the above notions into account, we can define the *Grid rendezvous* problem as follows.

Problem 1: (*Grid rendezvous*) Given a *rendezvous point* $\mathbf{q}_1$ and a group of N agents, design a control law that will drive all virtual grids associated with the agents so that they coincide at $\mathbf{q}_1$.

When considering a solution to Problem 1, we assume there is an obstacle-free circle segment centered at $\mathbf{q}_1$ containing the initial positions of all grids.

Once the agents have established a common formation representation by matching their respective virtual grid structures, this unified grid $\mathcal{G}$ can be used for coordinating formation transitions. The goal of this transition coordinator can be formalized as follows.

Problem 2: (*Transition coordination*) Design a coordination law that will ensure a safe and collision-free transition from any initial formation on the formation grid $\mathcal{G}$ to the desired formation specified by the binary matrix $\mathbf{F}^{(t)}$.

In our proposed solutions to Problems 1 and 2 that are presented in the remainder of this paper, we are assuming the existence of a centralized supervisor providing waypoint and formation references to the agent group. This supervisor does not perform any coordination functions, it merely notifies the group of robots of the task that they need to perform. Furthermore, we are assuming that the agents can communicate between each other without any restrictions. We are however considering the effects of temporary communication failures.

III. GROUP SYNCHRONIZATION

We propose a consensus-based group synchronization scheme as a potential solution to Problem 1. As an extension to the commonly considered rendezvous problem, as described for instance in [7], [8], it is not only required that the virtual formation grids converge to the same point, but also that they are aligned with each other. Agents can initially be placed arbitrarily in the obstacle-free part of the environment. As they converge at the rendezvous point, they should also establish and maintain a common representation of group state through their respective *formation grids*.

An abstract grid structure is attached to every agent, serving as a virtual formation leader A supervisory controller provides waypoint references that the group should drive to. Additionally, it provides formation setpoints, specifying the position of each agent on the formation grid.

A. The non synchronized case

Simply by providing the same waypoint reference to all the agents, we will force their respective grids to meet at that point. However, as illustrated by the following example, this will not suffice to align the grids with each other and the agents will not achieve the desired formation.

Example 1: (*Non synchronized rendezvous*) From initial positions specified in Table I agents are sent to the rendezvous point $q_1 = [15.0 \; 2.0]^T$. Each agent is moving

[1]This corresponds to standard matrix notation.

a_n	$x_{gn}(0)$	$y_{gn}(0)$	$\theta_{gn}(0)$	$i_n(0)$	$j_n(0)$
a_1	-1.0	10.0	$6\frac{\pi}{4}$	6	1
a_2	0.0	1.0	$-\frac{\pi}{3}$	5	4
a_3	2.0	-7.0	0.0	6	7

TABLE I: Initial positions for Examples 1 and 2

independently, without exchanging any information with other agents.

Initial and final agent configurations are shown in Fig. 3a and 3b respectively. While the positions of grid centers converge to the rendezvous point, grid headings are not aligned, because each robot arrives from a different angle. As a result, the agents have not achieved the desired formation.

B. *Consensus-based group synchronization*

As Example 1 indicates, an additional control signal is required to align formation grid orientations at the rendezvous point. In the proposed controller, this signal is derived from the arithmetic mean of individual grid representation coordinates,

$$\begin{aligned}\overline{x}_g &= \frac{1}{N}\sum_{n=1}^{N} x_{gn}\\ \overline{y}_g &= \frac{1}{N}\sum_{n=1}^{N} y_{gn}\\ \overline{\theta}_g &= \frac{1}{N}\sum_{n=1}^{N} \theta_{gn}.\end{aligned} \tag{2}$$

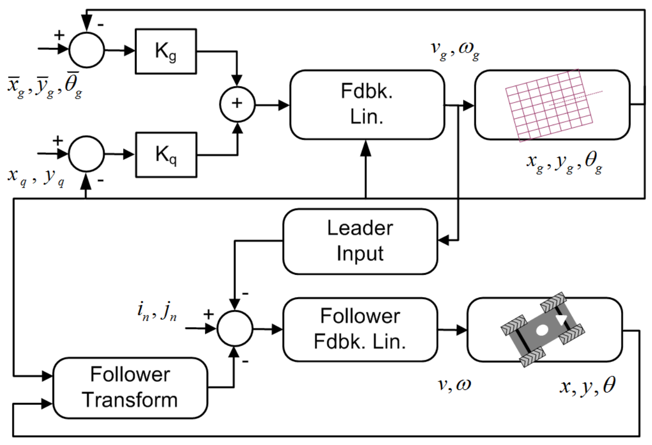


Fig. 1: Block diagram of the agent controller.

The controller structure for an individual agent is depicted in Fig. 1. Each agent is tracking its own virtual grid model at the specified angle and distance, using the leader-follower feedback linearization controller from [11]. The tracking angle and distance are computed from the formation specification $\mathbf{a}_n^G = [i_n\ j_n]^T$, relative to the top of the grid, and taking into account the number of cells N and real cell length L. Grid dynamics are modeled by a first-order unicycle model, described by the set of equations (2). A feedback linearization controller drives the grid model towards the given waypoint.

Inputs into the nonlinear grid controller are the superposition of two error signals. The waypoint error signal is used to drive the grid towards the current waypoint. The group error signal is responsible for achieving formation convergence. By adjusting the gain ratio on the two errors, we can influence how fast the group converges to the desired formation. Giving more weight to the waypoint error will yield a faster approach to the waypoint, but also slower group convergence, because agents lagging behind the group will need more time to catch up.

Example 2: (*Consensus-based coordination*) As in the previous example, agents are driven from their initial positions specified in Table I to the rendezvous point $q_1 = [15.0\ 2.0]^T$. However, this time the agents are communicating with each other to establish a common virtual grid representation. The control structure depicted in Fig. 1 is used to align them with the common grid.

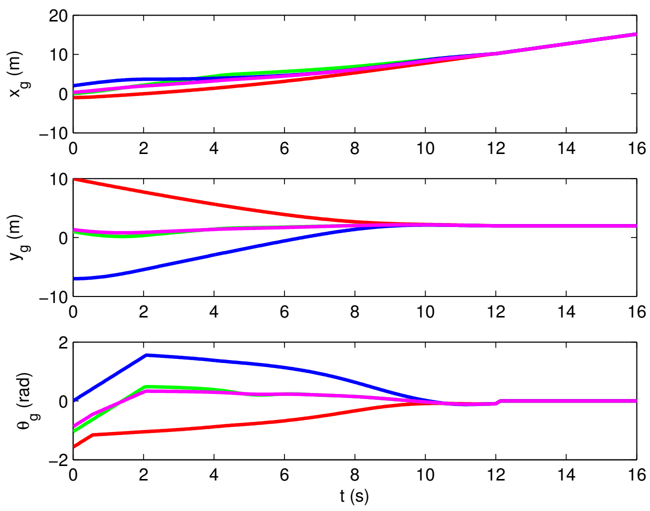
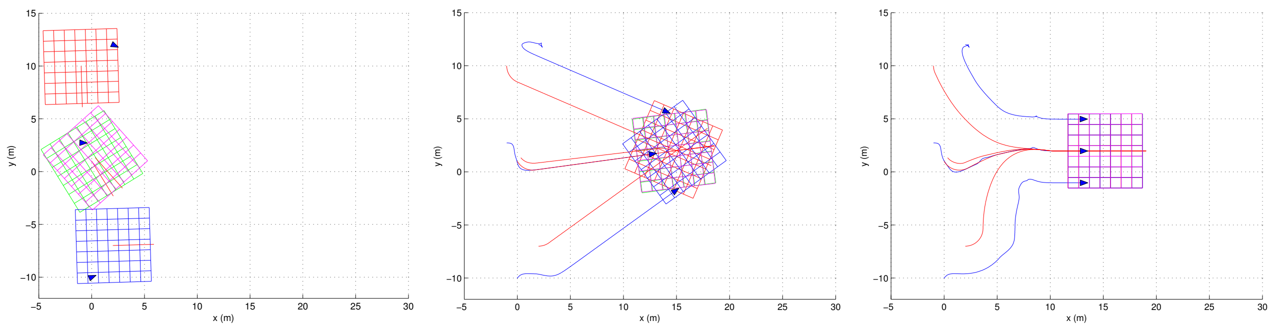


Fig. 2: Rendezvous with consensus-based synchronization, grid positions and orientations.

The initial and final agent configurations are shown in Fig. 3, and the evolution of grid coordinates over time is plotted in Fig. 2. It can be observed that all the virtual grids reach the rendezvous point simultaneously, with aligned headings. The agents have achieved the desired formation. Furthermore, because the aligned grids form a common representation of group state, they can be used to coordinate formation changes, as described in the following section.

IV. COOPERATIVE FORMATION RECONFIGURATION

Once the agent group has established a common representation of the formation, they can use the common grid abstraction to synchronize formation transitions. The synchronization scheme is based on our previous work described



(a) Initial agent positions with their respective grids.
(b) Final agent and grid positions, the non-coordinated case.
(c) Final agent and grid positions, the coordinated case.

Fig. 3: Rendezvous with and without grid consensus.

in [10]. However, in this paper we extend those results by using distributed computation and communication instead of a fully centralized approach. In the work presented here, we still assume the existence of a centralized coordinator, but its role is reduced to providing waypoint and formation setpoints.

Having received the desired formation setpoints, the algorithm performed by each agent can be outlined by these steps:

1) Target assignment
2) Path execution
    - Priority negotiation
    - Target re-assignment

The algorithm is discretized in space and in time. Space discretization implies that agent positions are only described by grid cell occupations. Time discretization means that agents move from one cell to the next at discrete time steps which are synchronized among the whole group. In the first phase, the agents negotiate to determine which position in the target formation is going to be assigned to which agent. Having determined their respective targets, the agents start moving towards them using the shortest paths. In case of conflict, when two agents require the same cell for their next move, priority negotiation takes place in order to prevent a collision. If an agent's path gets blocked by another agent that has already reached its target, they swap target positions.

A. Target assignment

Target assignment takes place after the supervisory group controller broadcasts the desired formation configuration to all agents. As indicated in [10], optimal target assignment is a fundamental prerequisite for the convergence of the entire formation transition algorithm. In our previous work, we used the Kuhn-Munkres assignment algorithm. To the best of our knowledge, no algorithm is currently known that can guarantee optimal assignment without considering all the initial states and all targets. For this reason, we require that every agent broadcast its computed distances to all the targets to all other agents. This information can be propagated through the agent group using a variation of the *flooding* algorithm described in [7].

Algorithm 1: Reconfiguration algorithm for agent a_m

```
   Input: $\mathbf{a}_m^G = [k_m \; l_m]^T$, $\mathbf{F}_{(t)} = [f_{ij}]$,$N$
1  tList $= \left\{\mathbf{t}_n | \mathbf{t}_n = [i_n \; j_n]^T \Leftrightarrow f_{ij} = 1\right\}$
2  foreach $\mathbf{t}_n \in$ tList do
3      $d_{mn} = |k_m - i_n| + |l_m - j_n|$
4  end
5  initialize distance matrix $\mathbf{D} = \mathbf{d}_m$
6  while numrows($\mathbf{d}_m$) $< N$ do
7      broadcast $\mathbf{D}$ to all agents
8      receive $\{\mathbf{d}_j | j \neq m\}$ and update $\mathbf{D}$
9  end
10 determine assigned target $\mathbf{t}_m$ from $\mathbf{D}$
11 find direct route to target $\left\{\mathbf{a}_m^G(k) | k = 1, \ldots d_{mm}\right\}$
12 $k = 1$
13 while new $\mathbf{F}^{(t)}$ not received do
14     broadcast $\mathbf{a}_m^G(k)$ to all agents
15     receive cells required by others, $\left\{\mathbf{a}_n^G(k) | m \neq n\right\}$
16     if $\mathbf{a}_m^G(k) \neq \mathbf{a}_n^G(k) \; \forall n \neq m$ then
17         move to $\mathbf{a}_m^G(k)$, $k =$min($k+1, d_{mm}$)
18     else
           /* Conflict, negotiate priority.
              */
19         cList $= \left\{a_n | \mathbf{a}_m^G(k) = \mathbf{a}_n^G(k)\right\}$
20         $\left(\left\{\mathbf{a}_m^G(k)\right\}, k\right)$ =negpri($\left\{\mathbf{a}_m^G(k)\right\}, k,$cList)
21     end
22 end
```

After receiving the desired formation matrix $\mathbf{F}^{(t)}$, each agent computes its distance to all target positions. The matrix $\mathbf{F}^{(t)}$ is a binary matrix, where the position of target t_n the desired formation is denoted by element at (k_n, l_n) having the value 1. If agent a_m has current coordinates (i_m, j_m) on the formation grid, then its distance to the targets is computed using the $\mathcal{L}_1$ norm,

$$d_{mn} = |i_m - k_n| + |j_m - l_n| \, . \tag{3}$$

Each agent broadcasts a row-vector $\mathbf{d}_m$ where each column represents its distance to the respective targets. It also keeps

Algorithm 2: Priority negotiation for agent a_m

```
Input: $\mathbf{a}_m(k)$,$\mathbf{t}_n$,$k$,cList
Output: $\mathbf{t}_n$,$\{\mathbf{a}_m(k)\}$,$k$
1  $d_m(k) = \|\mathbf{a}_m(k) - \mathbf{t}_m\|$
2  $d_{MAX} = d_m(k)$, $swapidx = 0$
3  foreach $a_n \in$ cList do
4      get $d_n(k)$ from $a_n$
5      if $d_n(k) > d_{MAX}$ then
6          if $d_m(k) = 0$ then
7              $d_{MAX} = d_n(k)$
8              $swapidx = n$
9          else
               /* Must yield to $a_n$ */
10             break
11         end
12     end
13     if $d_n(k) = 0$ then
14         $swapidx = n$
15     end
16 end
17 if $swapidx \neq 0$ then
       /* Swap targets. */
18     if $d_{MAX} = d_m(k)$ then
19         Claim other agent's target.
20     else
21         if $d_m(k) = 0$ then
22             Yield target to other agent.
23         end
24     end
25 else
       /* $a_m$ has longest path. */
26     $k = k + 1$
27 end
```

receiving distance-to-targets information from other agents, until it can assemble the whole distance matrix $\mathbf{D} = [d_{mn}]$. Then each agent runs the optimal assignment algorithm to determine its assigned location in the target formation. The current implementation is rather inefficient, because all the agents are running the same $O\left(n^3\right)$ algorithm. However, because the convergence of the whole formation transition procedure depends on the optimality of the assignment, this is the only viable solution at this point. Improvements to the formation transition algorithm are a topic of ongoing research.

B. Path execution

Once each agent has determined its target position in the new formation, they need to compute the transition paths. If the initial position of agent a_n has the coordinates $\mathbf{a}_n = [i_0\ j_0]^T$ and its target position is $\mathbf{t}_n = [i_t\ j_t]^T$, then its transition path consists of the set of cells

$$\begin{aligned}\mathcal{P} = &\left\{[i\ j]^T \,|i = i_0, j = j_0, \dots j_t\right\} \\ &\cup \left\{[i\ j]^T \,|i = i_0, \dots i_t, j = j_t\right\}. \end{aligned} \tag{4}$$

The agents move on $\mathcal{L}_1$ shortest paths on the formation grid, first along the rows and then along the columns.

Before each move, every agent broadcasts the coordinates of the next cell along its path to all its neighbors and waits to receive their replies. If there is no other agent requiring the same cell for its next move, it starts executing the motion immediately. If there is a conflict, a priority negotiation procedure is initiated. When none of the agents in conflict has reached its target position yet, priority is given to the agent with the longest remaining path. If the cell in conflict is a target cell already occupied by an agent, that agent swaps paths with the one having the longest remaining path. Once an agent has reached its destination, it keeps listening to move broadcasts by other agents, ready to swap paths with the ones that are coming its way.

The complete reconfiguration procedure is outlined by Algorithm 1 and Algorithm 2. Algorithm 1 describes the reconfiguration sequence performed by each agent. Algorithm 2 shows the priority negotiation procedure.

V. GROUP NAVIGATION SIMULATIONS

The simulations in this section are implemented using MATLAB$^{\text{TM}}$and Simulink®. Discrete-event control logic is implemented using Stateflow®.

A. Rendezvous and reconfiguration

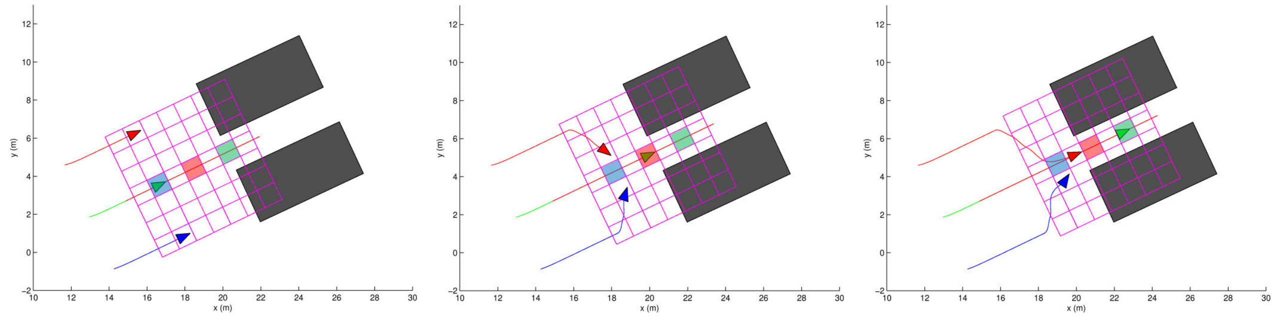
This simulation integrates the group synchronization controller described in Section III with the formation reconfiguration algorithm from Section IV. The agents start from initial positions given in Table I, meet at the rendezvous point $\mathbf{q}_1$ and continue towards waypoint $\mathbf{q}_2$. To avoid the obstacle between $\mathbf{q}_1$ and $\mathbf{q}_2$ they change their formation along the way.

System behavior during rendezvous has already been discussed in Section III. Snapshots from the reconfiguration sequences and agent trajectories are shown in Fig. 4. Agents are advancing through the common grid representation cell by cell, thus avoiding collisions. In Fig. 4b agents a_1 and a_3 negotiate for priority and a_3 yields, because a_1 has a longer path to its target.

B. Temporary communication failure

In this simulation we demonstrate group behavior under temporary communication failure. This is simulated by discarding all incoming and outgoing information for agent a_3. This includes information exchanged with other agents, as well as information received from the supervisor. The simulation shows that the control algorithm can enable the group to realign and restore the formation after communication has been reestablished.

This simulation is a continuation of Example 2. Only the y-positions and orientations are plotted in Fig. 5, because



(a) $\mathbf{a}_1^G = (6,1)$, $\mathbf{a}_2^G = (6,4)$, $\mathbf{a}_3^G = (6,7)$ (b) $\mathbf{a}_1^G = (6,3)$, $\mathbf{a}_2^G = (4,4)$, $\mathbf{a}_3^G = (6,5)$ (c) $\mathbf{a}_1^G = (5,4)$, $\mathbf{a}_2^G = (2,4)$, $\mathbf{a}_3^G = (6,5)$

Fig. 4: Formation reconfiguration snapshots. Shaded grid cells denote target positions.

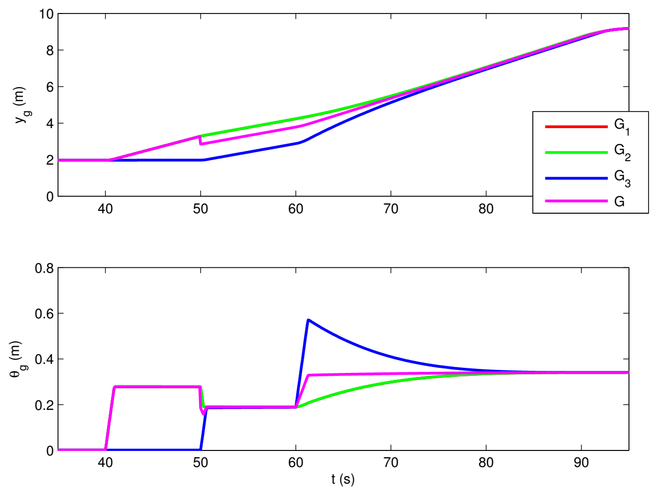


Fig. 5: Formation evolution with communication failure for a_3 between $t = 35$s and $t = 50$s.

no significant details are visible in the x-position signal. At $t = 30$s the formation has been established and the group is moving towards the waypoint $\mathbf{q}_2 = [30.0\ 2.0]$. At $t = 35$s agent a_3 looses contact with the group, and at $t = 40$s the supervisor broadcasts the new waypoint coordinates, $\mathbf{q}_3 = [40.0\ 9.0]$. Agent a_3 is not aware of the waypoint change and starts diverging from the group. At $t = 50$s communication is reestablished a_3 starts moving to rejoin the group. The collective grid position estimate $\mathcal{G}$ has a discontinuity at $t = 50$s because at that point agents a_1 and a_2 start getting fresh data from a_3 again and including it into the group state estimate.

VI. CONCLUSIONS AND FUTURE WORKS

We have presented a control scheme that enables a team of agents to converge to a rendezvous point, establish a common, grid-based representation of group state, and use this representation to switch between arbitrary formations. The formation switching algorithm prevents inter-agent collisions during formation changes. All computations are performed locally by the agents themselves, and the role of the centralized supervisor is reduced to providing waypoints and formation setpoints. The motion controllers take into account non-holonomic constraints of agent kinematics.

Simulations presented to demonstrate the validity of the proposed approach are preliminary proof-of-concept results. Much work remains to be done in order to refine and validate the described methodology. A thorough stability analysis must be performed for the consensus-based group synchronization controller. Similarly, the correctness of the distributed reconfiguration algorithms must be formally analyzed. The possibility to refine the algorithms and reimplement them using only nearest neighbor information is also being considered. Finally, the algorithms should be validated experimentally on actual robotic platforms.

REFERENCES

[1] T. Balch and R. Arkin, "Behavior-based formation control for multi-robot teams," *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 6, pp. 926–939, Dec 1998.

[2] N. Michael and V. Kumar, "Planning and Control of Ensembles of Robots with Non-holonomic Constraints," *The International Journal of Robotics Research*, vol. 28, no. 8, p. 962, 2009.

[3] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: algorithms and theory," *Automatic Control, IEEE Transactions on*, vol. 51, no. 3, pp. 401–420, March 2006.

[4] J. Hendrickx, B. Fidan, C. Yu, B. Anderson, and V. Blondel, "Formation reorganization by primitive operations on directed graphs," *IEEE Transactions on Automatic Control*, vol. 53, no. 4, pp. 968–979, May 2008.

[5] C. Belta and V. Kumar, "Abstraction and control for groups of robots," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 5, pp. 865–875, Oct. 2004.

[6] R. Murray, "Recent research in cooperative control of multivehicle systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, p. 571, 2007.

[7] F. Bullo, J. Cortés, and S. Martínez, *Distributed Control of Robotic Networks*, ser. Applied Mathematics Series. Princeton University Press, 2009, electronically available at http://coordinationbook.info.

[8] W. Ren, R. Beard, and E. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Systems Magazine*, vol. 27, no. 2, pp. 71–82, 2007.

[9] D. Miklic, S. Bogdan, and R. Fierro, "A grid based approach to formation reconfiguration in cluttered environments," in *European Control Conference, ECC*. Budapest, Hungary: EUCA, August 2009.

[10] D. Miklic, S. Bogdan, R. Fierro, and S. Nestic, "A discrete grid abstraction for fromation control in the presence of obstacles," in *Internatonal Conference on Intelligent Robots and Systems, IROS*. St. Louis, MO, USA: IEEE/RSJ, October 2009, (to appear).

[11] A. Das, R. Fierro, V. Kumar, J. Ostrowski, J. Spletzer, and C. Taylor, "A vision-based formation control framework," *IEEE Trans. on Robotics and Automation*, vol. 18, no. 5, pp. 813–825, October 2002.