2015 American Control Conference
Palmer House Hilton
July 1-3, 2015. Chicago, IL, USA

Distributed Virtual Leader Moving Formation Control Using Behavior-based MPC

Greg Droge

***Abstract*—In this paper we present a virtual leader formation control algorithm in which agents collaboratively adapt the formation and its motion in a distributed fashion. The formation is defined in terms of scaling and rotation parameters as well as agent offsets from of a virtual leader. The leader is virtual in that it is purely an artifact of a distributed optimization framework. Adaptation of the formation and control of the virtual leader is achieved through an MPC framework, which allows individual dynamic constraints to be considered in the optimization. By also defining the virtual leader motion in terms of a parameterized controller, distributed parameter optimization techniques are employed to solve the optimization problem at each iteration of the MPC algorithm. This allows agents to respect the underlying network constraints, forming a distributed implementation which scales well to large numbers of agents.**

I. INTRODUCTION

Formation control is a canonical problem for multi-agent control approaches as it requires the tight coupling of agent motion at a local level to achieve a desirable collective outcome. Maintaining formation while moving requires even tighter coupling of motion and decision-making. Not only must each agent converge to a particular position to form the formation, but they must coordinate how they move to stay in formation. This paper presents an approach for formation control which leverages the reactive elements of feedback control and the adaptive element of an optimization framework through the use of distributed optimization.

The presented approach builds upon three classes of moving formation control. The first class consists of local or relative approaches where formation control is defined in terms of highly reactive control laws defined between neighboring agents (see, for example, [1], [2], [3] and the references therein). The second class builds upon the first by introducing a centralized computing component in the form of a virtual leader to adjust a leader's motion based on how well agents are tracking their desired positions, e.g. [4], [5]. The third class is distributed model predictive control (MPC) techniques where at each step in time, agents choose their control by collaboratively minimizing a cost which simulates their actions into the future, e.g. [6], [7], [8].

While each class has its benefits, there are drawbacks to each approach. The relative approaches often require an a-priori analysis of the network to ensure the structure of the formation is maintained, e.g. [1], [9]. Both the local and virtual leader approaches depend upon predefined control laws which may not be highly adaptable, even though they are highly reactive, e.g. [6]. Additionally the virtual leader approaches have a centralized component, which may degrade performance with large numbers of agents [5]. While the MPC frameworks are highly adaptable and distributed, they require the shuffling of large amounts of information in order to optimize their trajectories. This optimization may affect reaction time to unforeseen events and an "emergency" control is often defined to ensure a reactive element [6].

The approach taken in this paper is to combine benefits from the relative approach with those of MPC by taking a behavior-based MPC approach, e.g. [10], [11], [12]. A parameterized feedback control is used in conjunction with distributed optimization over the parameters to achieve both a reactive and adaptive moving formation which is not burdened by excessive communication. The feedback control to be executed by the individual agents will be developed in Section II. Then, in Section III, the interactions between agents and the control of the collective formation using distributed MPC will be discussed. The chapter will end with an example in Section IV and concluding remarks in Section V.

II. ARC-BASED LEADER-FOLLOWER FORMATION CONTROL

A key component to the formulation of the virtual leader formation control is the control law which allows agents to follow the virtual leader. Thus, this section will focus on the development of a controller to follow a virtual leader while Section III will focus on the collaboration between agents to control the virtual leader and adapt the formation.

A. Formation Definition and Tracking

The formation is constructed by defining the desired follower position, q_{f_d}, as the point τ_f in the coordinate

Email: greg.droge@navy.mil.
SPAWAR Systems Center Pacific, San Diego, CA 92152, USA.

978-1-4799-8684-2/$31.00 ©2015 AACC

frame with origin at the leader position, q_l, and x-axis pointing with the orientation of the leader, ψ_l. Similar to [13], we also allow the formation to adapt by having an arbitrary rotation, Ψ, about the leader and an arbitrary scaling parameter, $\gamma > 0$, which can be used to shrink or expand the formation.

In a fixed global frame, the desired follower position can be expressed as

$$q_{f_d} = R(\psi_l + \Psi)\gamma\tau_f + q_l, \tag{1}$$

where $R : \mathbb{R} \longrightarrow \mathbb{R}^{2\times 2}$ is a rotation matrix.

To develop a controller for an agent to track q_{f_d}, we break down the problem into four steps.

1) The leader motion is defined.
2) The motion of q_{f_d} is extrapolated.
3) The motion of a point on the follower that does not have a nonholonomic constraint is defined.
4) A controller is developed.

1) Leader Motion: To allow agents to have a simple, yet expressive, method to control the leader, the leader will execute arc-based motions[1]. This is achieved by modeling the leader with a unicycle motion model executing constant translational and rotational velocities, denoted as v_l and ω_l respectively. The unicycle motion model is defined using a unit vector $h_\psi = [\cos(\psi),\ \sin(\psi)]^T$, to denote the direction of motion. The leader motion can then be written as:

$$\begin{bmatrix}\dot{q}_l \\ \dot{\psi}_l\end{bmatrix} = \begin{bmatrix} v_l h_{\psi_l} \\ \omega_l\end{bmatrix}. \tag{2}$$

As depicted in Figure 1, executing constant v_l and ω_l corresponds to the leader executing a circular motion. The radius of the circle can be written as $\rho_l = \frac{v_l}{|\omega_l|}$ for $\omega_l \neq 0$. The center, c, can be directly extracted evaluating the solution to (2) assuming constant velocities and can be expressed as

$$c = \frac{v_l}{\omega_l} J h_{\psi_l} + q_l,\ J = \begin{bmatrix} 0 & -1 \\ 1 & 0\end{bmatrix}. \tag{3}$$

2) Desired Follower Motion: With the leader motion defined, the desired motion of the follower agent can be extracted. The motion is given in the following Lemma:

Lemma II.1. *Assuming the formation parameters (γ, Ψ) and the leader motion parameters (v_l, ω_l) are fixed, the desired follower motion can be given as:*

$$\dot{q}_{f_d} = v_{f_d} h_{\psi_{f_d}} \tag{4}$$

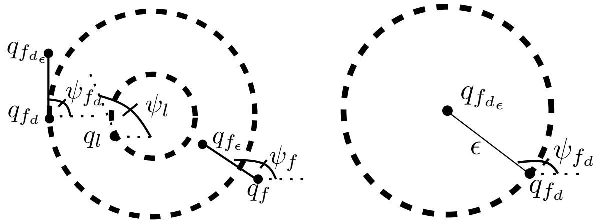

Fig. 1. On the left is shown a diagram of positions: the leader position, q_l, and orientation, ψ_l, the actual and desired follower positions, q_f and q_{f_d}, and orientations, ψ_f and ψ_{f_d}, as well as the actual and desired follower ϵ-points, q_{f_ϵ} and $q_{f_{d_\epsilon}}$. The right image shows the set of all positions that could satisfy $q_{f_\epsilon} = q_{f_{d_\epsilon}}$.

where

$$\psi_{f_d} = \begin{cases} \psi_l & \omega_l = 0 \\ -sign(\omega_l)\frac{\pi}{2} + \phi_{f_d} & otherwise \end{cases}$$
$$v_{f_d} = \begin{cases} v_l & \omega_l = 0 \\ \rho_{f_d}|\omega_l| & otherwise \end{cases}, \tag{5}$$

Proof: The proof falls out by considering the fact that q_{f_d} is fixed in the leader frame and examining the geometry with respect to the circular motion of the leader. ■

3) Motion of Desired ϵ-point: A basic concept presented in [17] is that while the center of the robot has a non-holonomic constraint, a point directly in front of the center of the robot has no such constraint. We thus define and control the motion of a point directly in front of the desired position.

We refer to this point as the desired ϵ-point and denote it as $q_{f_{d_\epsilon}}$. It can be expressed in terms of q_{f_d} and ψ_{f_d} as:

$$q_{f_{d_\epsilon}} = q_{f_d} + \epsilon h_{\psi_{f_d}}. \tag{6}$$

Noting that $\dot{h}_\psi = \omega J h_\psi$, the desired motion of the ϵ-point can be written as:

$$\dot{q}_{f_{d_\epsilon}} = v_{f_d} h_{\psi_{f_d}} + \epsilon\omega_{f_d} J h_{\psi_{f_d}}. \tag{7}$$

Also, note that $\ddot{q}_{f_{d_\epsilon}}$ is used in a proof of convergence, so (7) can be differentiated to obtain:

$$\ddot{q}_{f_{d_\epsilon}} = v_{f_d}\omega_{f_d} J h_{\psi_{f_d}} - \epsilon\omega_{f_d}^2 h_{\psi_{f_d}}. \tag{8}$$

4) ϵ-Point Control: We are now ready to define the desired control law to be used in the leader-follower behavior. We again use the unicycle motion model to model the follower agent. Denoting the follower position as q_f, orientation as ψ_f, and orientation vector as $h_{\psi_{f_d}}$, the ϵ-point can be written as:

$$q_{f_\epsilon} = q_f + \epsilon h_{\psi_f}.$$

[1] In a very real sense, this is a gradient-based, multi-agent extension of the celebrated Dynamic Window Approach (DWA), [10]. DWA uses arc-based motions to control a single agent and has been widely used for its ability to effectively address the non-holonomic constraints of wheeled vehicles, e.g. [14], [15]. It even forms the default local planner for the increasingly popular Robot Operating System [16].

To control the ϵ-point, the following control law is employed:

$$\begin{bmatrix} v_f \\ \omega_f \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{\epsilon} \end{bmatrix} \begin{bmatrix} \cos(\psi_f) & \sin(\psi_f) \\ -\sin(\psi_f) & \cos(\psi_f) \end{bmatrix} \dot{q}_{f_\epsilon}, \quad (9)$$

where

$$\dot{q}_{f_\epsilon} = \dot{q}_{f_{d_\epsilon}} + k_p(q_{f_{d_\epsilon}} - q_{f_\epsilon}), \quad (10)$$

and $k_p > 0$ is a constant. A Lemma is now given on the convergence of (9):

Lemma II.2. *Assuming the formation parameters (γ, Ψ) and the leader motion parameters (v_l, ω_l) are fixed, the control law given in (9) will cause q_{f_ϵ} to approach $q_{f_{d_\epsilon}}$ at an exponential rate.*

Proof: This is a slightly simplified form of the controller in [17] and the proof of convergence can be followed exactly. ∎

B. Perfect Tracking Using Approximate Control

As mentioned in Lemma II.2, the controller in (9) will allow q_{f_ϵ} to track $q_{f_{d_\epsilon}}$. A theorem is now given to show that, due to the motion of the follower and leader, the follower can actually perfectly q_{f_d}.

Theorem II.3. *Assuming the formation parameters (γ, Ψ) and the leader motion parameters (v_l, ω_l) are fixed, the error $e = q_f - q_{f_d}$ has an asymptotically stable equilibrium at $e = 0$.*

Proof: As the robot achieves ϵ-tracking, the set of possible positions of the robot at time t forms a circle of radius ϵ around the point $q_{f_\epsilon}(t)$, as depicted in Figure 1. The worst possible position at any given time for the robot would be a distance of 2ϵ from the desired point. But $q_{f_d}(t)$ also exists on the circle, making it possible that $q_f(t) = q_{f_d}(t)$.

Lemma II.2 states that the position of the robot will converge to the circle with its orientation pointed towards the center. Thus, we need only to evaluate the orientation of the robot in the constraint set. The reason being that each value of $\psi_f \in [-\pi,\ \pi)$ will correspond to a unique point on the circle. So if $\psi_f(t) \longrightarrow \psi_{f_d}(t)$ then we know that $q_f(t) = q_{f_d}(t)$. Therefore, we evaluate the following candidate Lyapunov function:

$$V = -h_{\psi_f}^T h_{\psi_{f_d}} + 1, \quad (11)$$

which is zero for $\psi_f(t) = \psi_{f_d}(t)$ and positive for $\psi_f(t) - \psi_{f_d}(t) \in (-\frac{\pi}{2},\ \frac{\pi}{2})$. Taking the time derivative of V we obtain:

$$\begin{aligned} \dot{V} &= -\dot{h}_{\psi_f}^T h_{\psi_{f_d}} - h_{\psi_f}^T \dot{h}_{\psi_{f_d}} \\ &= -\frac{v_{f_d}}{\epsilon}(h_{\psi_f}^T J^T h_{\psi_{f_d}})^2 + \quad (12) \\ &(1 - h_{\psi_f}^T h_{\psi_{f_d}})\omega_{f_d} h_{\psi_f}^T J^T h_{\psi_{f_d}} \end{aligned}$$

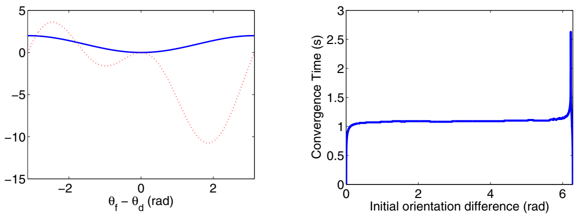

Fig. 2. The left show the Lyapunov function, V, given in (11) for $v_{f_d} = .5$, $\epsilon = .1$, and $\omega_{f_d} = 5$. V is plotted as a solid line and $\dot{V}$ is plotted as a dotted line. The right image shows convergence times for the values in the middle image for all possible deviations in orientation for $v_{f_d} = .5$, $\epsilon = .1$, and $\omega_{f_d} = 100$, showing that V may be conservative.

The first term is always negative, but the second term is not. However, we can capitalize on the fact that $h_{\psi_f}^T h_{\psi_{f_d}} \approx 1$ around $\psi_f = \psi_{f_d}$. So, for some region around $\psi_f = \psi_{f_d}$, $\dot{V} \leq 0$ with strict inequality for $\psi_f \neq \psi_{f_d}$. Thus, the $\frac{v_{f_d}}{\epsilon}$ term will dominate for some region around the origin and the bigger the value of $\frac{v_{f_d}}{\epsilon}$ in comparison to ω_{f_d}, the larger the region of convergence. ∎

Depicted in Figure 2 is the Lyapunov function V and its time derivative $\dot{V}$ for values of $v_{f_d} = .5$, $\epsilon = .1$, and $\omega_{f_d} = 5$. The region of convergence derived from V for the mentioned values is $\psi_f - \psi_{f_d} \in [-1.79,\ \pi)$. We note that the region of convergence determined by V also seems quite conservative. For the very extreme case of $\omega_{f_d} = 100$, $v_{f_d} = .5$, and $\epsilon = .1$, Figure 2 also depicts the time for convergence to within 1% of the desired value for $\psi_f - \psi_{f_d} \in [-\pi,\ \pi]$. Convergence is always achieved.

C. Obstacle Avoidance Control

While a cost for collisions will be included as part of the optimization, a cost barrier cannot be used for numerical reasons (discussed further in Section III). To ensure agents avoid obstacles, an additional term is added to the control in (10). For sake of brevity, it is not discussed further other than to mention that we use the vector-field approach given in [18].

III. Virtual Leader Behavior-based MPC

The control law developed in the previous section defines how agents will move with respect to a leader and the given formation parameters. This section discusses how agents collaborate to decide upon both the motion of the leader and the rotation and scaling of the formation.

The section begins by giving a brief introduction to the MPC scheme to be utilized. The specifics of the virtual leader formation control problem are then discussed, leading to a statement of the algorithm to be employed.

A final contribution of the section is the discussion of the cost to be minimized.

A. Behavior-based MPC

The idea behind behavior-based MPC is to adapt the parameters of the control online through a receding horizon approach, evaluating at each time step how the parameters will affect the robot for some time horizon into the future. This allows for highly reactive control laws to be used and MPC emplyed for adaptation without the need to communicate entire trajectories, only a number of parameters. For the multi-agent scenario, the cost minimization is the method by which agents are able to collaborate.

Before proceeding, two important items should be mentioned. First, as no agent's dynamics depend on any other agent, agent's are only required to communicate with agents upon which their cost depends [13]. The second aspect is that to use gradient-based distributed optimization, the cost must be defined such that the gradients are bounded [19]. This does not permit the use of cost barriers as a method to give guarantees (such as collision avoidance).

B. Virtual Leader Algorithm

To implement the virtual leader approach, agents need to communicate and negotiate over various parameters. The parameters to negotiate upon are those that define the structure and movement of the formation. Namely, agent i must maintain a vector of parameters[2], $\theta^i = \begin{bmatrix}\Psi^i & \gamma^i & v_l^i & \omega_l^i\end{bmatrix}^T$, where the superscript i denotes that the variable is maintained by agent i, Ψ and γ are the rotation and scaling of the formation, and (v_l, ω_l) is the velocity pair that defines the motion of the virtual leader.

Beyond the distributed optimization of the mentioned parameters, agents must also come to agreement about the initial position and orientation of the leader, q_{l0}^i and ψ_{l0}^i, respectively. While these could potentially be variables to be optimized, we found that simply performing consensus (e.g. [1]) over the initial conditions provided better results . Besides parameters to be negotiated, each agent must also communicate its initial state, which we denote as x_0^i.

One key note to make is that the optimization and execution is performed simultaneously. If the time that optimization is started is denoted as t, agents are optimizing over the parameters that will begin to be executed at time $t + \delta_{execute}$. This allows $\delta_{execute}$ to be chosen such that the optimization is able to converge (or reasonably close) before the parameters are employed. *We emphasize that for agents to move in unison, they must be in strict agreement on the virtual leader motion and the parameters influencing the structure of the formation.*

To account for the difference between variables being optimized and variables being executed, we use the added notation $\bar{\Psi}^i, \bar{v}_l^i, \bar{\omega}_l^i, \bar{q}_l^i, \bar{\psi}_l^i$ to denote the values actually being executed by the agent. Also, the double time index, $m^i(t_0; t)$, denotes agent i's opinion at time t of what the variable m will be at time t_0. The behavior-based virtual leader MPC algorithm can now be stated in Algorithm 1.

Algorithm 1 Behavior-based Virtual Leader MPC

```
1) Initialize $t_0 = t + \delta_{execute}$ where $t$ is current time.
2) Agents communicate parameters with neighboring agents:
    a) Formation structure and movement: $\Psi^i, \gamma^i, \omega_l^i$ (and optionally $v_l^i$).
    b) Virtual leader initial conditions: $q_l^i(t_0; t)$, $\psi_l^i(t_0; t)$.
    c) Agent initial conditions: $x_0^i = x^i(t_0; t)$.
3) Update parameters:
    a) Update $\Psi^i, \gamma^i, \omega_l^i$ using distributed optimization.
    b) Update $q_l^i(t_0; t)$, $\psi_l^i(t_0; t)$ using consensus.
4) Repeat steps 2 and 3 until $t = t_0$.
5) Set $\bar{\Psi}^i = \Psi^i, \bar{v}_l^i = v_l^i, \bar{\omega}_l^i = \omega_l^i, \bar{q}_l^i = q_l^i, \bar{\psi}_l^i = \psi_l^i$.
6) Repeat steps 1 through 5.
```

C. Cost Definition

An essential aspect to maintaining the structural integrity and desirable movement of the formation comes through the definition of the cost. The receding horizon cost utilized in Section IV can be decomposed into four components[3], where the first is evaluated over the entire horizon and the latter three are evaluated at the end of the horizon:

1) Penalize proximity to obstacles

[2] We note that, while the network of agents can find desirable values for v_l and ω_l in the absence of obstacles, in the presence of obstacles, a very undesirable local minimum exists. Namely, to avoid obstacles, the agents can simply stop moving. Thus, in the examples in Section IV-B, v_l is held constant and the agents solely steer the virtual leader using ω_l.

[3] Many more costs are conceivable, such as costs on velocities, and positions, etc. In particular, in the absence of obstacles, we found that the cost $\frac{1}{2}||q_{il} - q_{goal}||^2$ was particular useful in helping to regulate v_l so that the leader agent converged to the goal location, q_{goal}. In the presence of obstacles, we found a constant leader velocity and the mentioned orientation cost to work quite well in guiding the virtual leader.

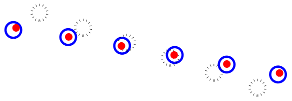

Fig. 3. This figure shows the actual positions of the agents plotted as circles with solid lines, the desired positions using the global variables as circles with dotted lines, and the desired positions using the process in Section IV-A as solid circles.

2) Penalize deviation from desired scaling
3) Penalize deviation between q_f and q_{f_d}
4) Penalize orientations not facing the goal location

Each cost is weighted by a value ρ_k, where k corresponds to the cost number above. The exact form of the costs can be found in Chapter 7.3.3 of [20].

IV. Results

In this section we present results of having agents travel through an obstacle field while maintaining formation. However, evaluating performance is not completely straight forward. Figure 3 shows an example where agents have deviated significantly from their desired positions, and yet they are clearly maintaining their line formation. Therefore, we first discuss a method of measuring the performance of the collective whole and then proceed to give the examples.

A. Measuring Formation Performance

To compare results from different trials, it is important to measure how well the agents maintain formation. However, as we allow agents to freely adapt the rotation and scaling of the formation, the sum of distances from agent's to their desired position as defined by the formation parameters and leader state is not a good measure. An example is shown in Figure 3 where teh agents are far from their desired positions, but definitely still executing the desired line configuration.

To evaluate the performance of the formation control we perform post-processing on their positions at each step in time. First, the best fitting scaling, rotation, and formation center is found in terms of the median sense (to remove the affect of agents which are out of formation). Then, the average and maximum squared error is calculated to evaluate the performance of the collective whole. Details of the exact calculation can be found in Chapter 7 of [20].

B. Virtual Leader Formation Results

To present an example of the virtual-leader behavior-based MPC framework, we present two sets of trials. The two sets consist of agents forming a line formation and a GT formation (similar to the Georgia Tech logo). In each trial, agents used distributed optimization to solve for ω_l, γ, and Ψ. The only centralized aspect of the trials was the occasional broadcast of a new goal position for the leader, enabling the agents to traverse the entire environment through a waypoint navigation approach. The line is used as it shows the ability for agents to maintain a formation without an underlying rigid structure. It is also highly rotationally variant, meaning as the line is rotated it can move very differently between obstacles. The GT formation presents an example showing the ability for agents to maintain arbitrary structures. It is different from the line formation in that the structure of the formation renders rotation much less useful.

For distributed optimization we used the method detailed in [13] which allows for time varying communication networks. It requires a gain on the gradient, k_g, and a proportional and integral gain on the disagreement, denoted as k_P and k_I, respectively. In each trial, the gains for distributed optimization were set as $k_g = .1$, $k_p = 2.5$, and $k_I = 1.5$. Together with the costs in Section III-C and a value of $\delta_{execute} = 0.4$, the distributed optimization had sufficient amount of time to achieve sufficient agreement. *We emphasize that the most important aspect for the agents to maintain the structure of the formation is for the disagreement between agents to be small.*

Each formation was run for two trials to illustrate the benefit of the optimization in maintaining the structure of the formation. The first trial consisted of the agents controlling only the leader velocity, without regard to the structure of the formation. This was achieved by setting $\rho_1 = \rho_2 = \rho_3 = 0$ and $\rho_4 = 1$. The second trial consisted of setting all gains to one. Basically a trial without tuning the costs, but taking deviation from formation and obstacle proximity into account.

The results are depicted in Figure 4. Both examples show the ability of the controller discussed in Section II to keep agents in formation. The trials used a value of $\epsilon = 0.1$, but the average distance from agents to the desired position is far less than ϵ. Without tuning the costs, a large reduction in both the maximum distance and average distance is attained.

V. Conclusion

In this paper, a distributed virtual-leader formation control was developed. Agents used a distributed MPC framework to collaboratively adapt the leader motion and parameters affecting the formation to navigate through an obstacle field. Two examples demonstrated the ability for agents to rotate and scale the formations to maintain the structure of the formation while avoiding obstacles. As

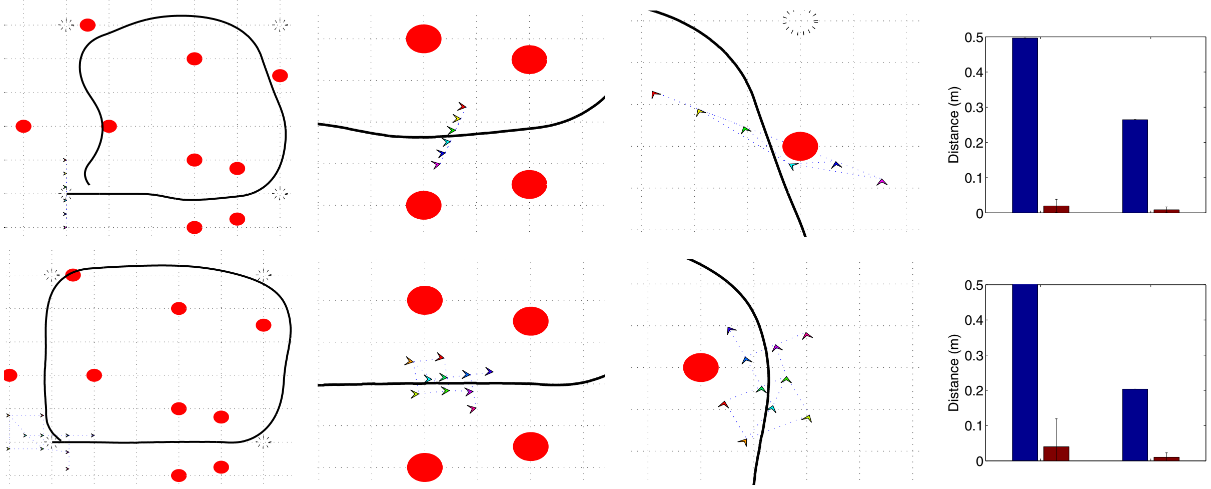

Fig. 4. This figure shows the results for the line formation on the top row and GT formation on the bottom row. The left most image shows the initial environment, nominal positions of the agents, and the path taken around the obstacles. The middle two images show the agents at different points in the environment. The agents are shown as triangles, the obstacles are shown as solid circles, the waypoints are shown as dotted circles, and the trajectory of the leader agent around the environment is plotted as a solid line. The right most image shows two groups of bars plotting the maximum and average distances (according to Section IV-A) for leader only control (i.e. $\rho_1 = \rho_2 = \rho_3 = 0$, $\rho_4 = 1$) in the left group and $\rho_k = 1$ on the right group. Note that for visualization, the maximum distance in the leader-only control for the line formation was cut off and actually has a value of 1.23.

the leader was a construction of distributed optimization, the presented algorithm allows agents to move about in formation without any one agent performing a significant amount of the computation.

References

[1] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton Univerisity Press, 2010.

[2] R. M. Murray, "Recent research in cooperative control of multivehicle systems," *Transactions-American Society of Mechanical Engineers Journal of Dynamic Systems Measurement and Control*, vol. 129, no. 5, p. 571, 2007.

[3] J. A. Gouvea, F. Lizarralde, and L. Hsu, "Formation control of dynamic nonholonomic mobile robots with curvature constraints via potential functions," in *American Control Conference (ACC)*. IEEE, 2013, pp. 3039–3044.

[4] B. J. Young, R. W. Beard, and J. M. Kelsey, "A control scheme for improving multi-vehicle formation maneuvers," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 2. IEEE, 2001, pp. 704–709.

[5] P. Ogren, E. Fiorelli, and N. E. Leonard, "Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment," *Automatic Control, IEEE Transactions on*, vol. 49, no. 8, pp. 1292–1302, 2004.

[6] T. Keviczky, F. Borrelli, and G. Balas, "Decentralized receding horizon control for large scale dynamically decoupled systems," *Automatica*, vol. 42, no. 12, pp. 2105–2115, 2006.

[7] C. Conte, N. R. Voellmy, M. N. Zeilinger, M. Morari, and C. N. Jones, "Distributed synthesis and control of constrained linear systems," in *American Control Conference (ACC)*. IEEE, 2012, pp. 6017–6022.

[8] C. Jones, Y. Pu, S. Riverso, G. Ferrari-Trecate, M. N. Zeilinger, *et al.*, "Plug and play distributed model predictive control based on distributed invariance and optimization," in *The 52nd Conference on Decision and Control*, 2013.

[9] T. Liu and Z.-P. Jiang, "A nonlinear small-gain approach to distributed formation control of nonholonomic mobile robots," in *American Control Conference (ACC)*. IEEE, 2013, pp. 3051–3056.

[10] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[11] T. Howard, C. Green, and A. Kelly, "Receding horizon model-predictive control for mobile robot navigation of intricate paths," in *Field and Service Robotics*. Springer, 2010, pp. 69–78.

[12] G. Droge and M. Egerstedt, "Distributed parameterized model predictive control of networked multi-agent systems," *American Control Conference (ACC)*, 2013.

[13] ——, "Proportional integral distributed optimization for dynamic network topologies," *American Control Conference (ACC)*, 2014.

[14] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous uav guidance," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, pp. 65–100, 2010.

[15] P. Ogren and N. Leonard, "A convergent dynamic window approach to obstacle avoidance," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 188–195, 2005.

[16] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2, 2009.

[17] R. Olfati-Saber, "Near-identity diffeomorphisms and exponential epsilon tracking and epsilon stabilization of first-order nonholonomic se(2) vehicles," in *Proceedings of the American Control Conference (ACC)*, vol. 6. IEEE, 2002, pp. 4690–4695.

[18] R. Arkin, *Behavior-based robotics*. The MIT Press, 1998.

[19] J. Wang and N. Elia, "Control approach to distributed optimization," in *48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2010, pp. 557–561.

[20] G. Droge, "Behavior-based model predictive control for networked multi-agent systems," Ph.D. dissertation, Georgia Institute of Technology, 2014.