A FORMATION CONTROL ALGORITHM USING VORONOI REGIONS

Magnus Lindhé [*] **Karl Henrik Johansson** [**]

[*] *Department of Automatic Control*
Royal Institute of Technology
SE-100 44 Stockholm, Sweden
lindhe@ee.kth.se
[**] kallej@ee.kth.se

Abstract: We present an algorithm for stabilizing a hexagonal lattice formation of autonomous robotic agents. The algorithm is decentralized and each agent only needs to detect the relative positions of its neighbors. By partitioning the plane into Voronoi regions we can guarantee collision safety, even when the algorithm is used to produce waypoints for a non-holonomic agent to follow. In each iteration every agent moves to the centroid of the vertices of its Voronoi region, which yields formation cohesion. We define asymptotic formation stability that is independent of rotation or translation of the whole formation and prove local asymptotic stability. Finally we present simulations that confirm the stability analysis and illustrate the use of the algorithm with car-like robots.

Keywords: Cooperative control, Multi-agent systems, Decentralized control

1. INTRODUCTION

Recent advances in fields such as microcontrollers, integrated radio circuits, ad-hoc networking protocols and sensors have made it feasible to manufacture small and cheap robots with many of the capabilities needed for autonomous operation. By using groups of such simple robots to perform tasks, several advantages can be gained. Robustness against vehicle failure is increased, the robots can help each other in rough terrain (Trianni *et al.*, 2005) and communications can be facilitated by relaying (Nguyen *et al.*, 2004). By moving in formation, the robots can form synthetic apertures for deep-space exploration (Scharf *et al.*, 2004) or sweep for mines (Healey, 2001).

Due to these advantages, the field of multi-agent formation control has been studied extensively. One approach is using artificial potentials (Leonard and Fiorelli, 2001), which yields decentralized control laws and provable formation convergence. Reynolds, (Reynolds, 1987), pioneered the branch that uses behavioral control, inspired by biological flocks and swarms. Others suggest considering the whole group as a virtual structure that can be controlled as a unit. The prescribed trajectory of each member is then communicated to local controllers that follow the trajectories (Beard *et al.*, 2001). As a final example, some use leader-follower architectures where one (possibly virtual) agent leads the whole group (Desai *et al.*, 1998) or the agents follow each other in some prescribed order (Liu *et al.*, 2003). When analyzing flocking methods, it is common to visualize the information interchange between agents as a (possibly directed) graph. Important properties of the resulting collective motions can then be derived by algebraic graph theory (Tanner *et al.*, To appear).

The contribution of this paper is a new algorithm using Voronoi regions for collision safety and formation cohesion. This gives an advantage when controlling non-holonomic platforms. As opposed to some of the above approaches, our algorithm not only computes a reference trajectory, but also an associated region

in which the agent can safely maneuver. This lends itself well to hierarchical architectures where a higher-level controller, assuming holonomic agents, produces waypoints for a lower-lever controller that is adapted to the actual platform dynamics. The algorithm is decentralized and scales well with group size.

Our approach is inspired by the work by Cortés *et al.* on optimally placing sensor networks using Voronoi regions (Cortés *et al.*, 2004). In earlier work (Lindhé *et al.*, 2005), we have described another algorithm that uses Voronoi regions, but with a computationally more demanding method that was also more difficult to analyze from a stability point of view.

This paper is organized as follows: In Section 2, we describe our agent model and state the objectives that we want to fulfill. In Section 3 we detail our proposed algorithm before studying convergence and safety in Section 4. The stability is analyzed in a simplified setting, but we simulate the more general situation in Section 5. We also present simulations with a more realistic car-like agent model. Finally we conclude the paper in Section 6.

2. PROBLEM DEFINITION

In this section we define the agent dynamics, state the objectives that we want to fulfill and give a formal definition of asymptotic formation stability. We consider a group of K autonomous agents, $k = 0, 1, \ldots, K-1$, positioned at $P = \{\mathbf{p}_k\}$. Throughout this paper, we will use boldface letters for vectors. The agents are equipped with a means of sensing the relative position of the other agents and obey the following simple dynamics in $\mathbb{R}^2$:

$$\mathbf{p}_k(t+1) = \mathbf{p}_k(t) + \mathbf{u}_k(t)$$

The problem we consider is that of finding a control law $u_k : N_k \to U$ that maps the positions of the neighbors N_k of each agent k to a control output. The set of neighbors will be formally defined below. When each agent applies the control law, we want the resulting global closed-loop system to fulfill the following objectives:

- The agents should never collide with each other.
- The formation in the shape of a hexagonal lattice should be asymptotically stable.
- The algorithm should be decentralized, and thus suitable for implementation with access only to local information.
- The net movement of the group should be externally controllable, in order to move the formation around obstacles and choose a goal.

Before defining the asymptotic formation stability, we need the concept of Voronoi regions.

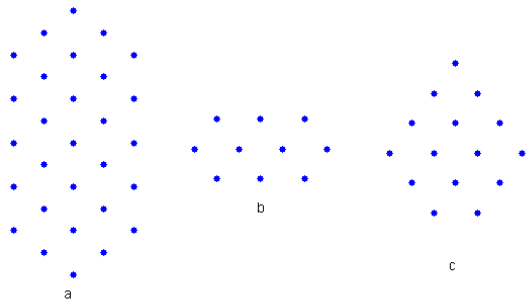

Fig. 1. Configurations a and b are hexagonal lattice formations, while in formation c, the top agent has only 2 neighbors, which is not allowed.

Definition 1. (Voronoi region and vertex). Let $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ be a set of points in $\mathbb{R}^2$. The *Voronoi region* $R_k(Z) \subset \mathbb{R}^2$ consists of all points that are closer to $\mathbf{z}_k$ than any other point in Z:

$$R_k(Z) = \{\mathbf{x} : |\mathbf{x} - \mathbf{z}_k| < |\mathbf{x} - \mathbf{z}_i| \ \forall\ i \neq k\}. \quad (1)$$

The line segments

$$\ell_k^i(Z) = \{\mathbf{x} : \ \exists i \neq k : |\mathbf{x} - \mathbf{z}_k| = |\mathbf{x} - \mathbf{z}_i| < |\mathbf{x} - \mathbf{z}_j| \ \forall\ j \neq i, k\} \quad (2)$$

constitute the boundary of $R_k(Z)$, i.e. $\partial R_k(Z) = \bigcup_i \ell_k^i$. The set of *Voronoi vertices* $V_k(Z)$ for $\mathbf{z}_k$ are the points

$$V_k^{ij}(Z) = \ell_k^i(Z) \bigcap \ell_k^j(Z). \quad (3)$$

We must also define the neighbor set of an agent:

Definition 2. (Neighbor set). The set of neighbors, N_k, to agent k consists of all agents that share a Voronoi vertex with agent k. The set of *close neighbors*, $C_k \subset N_k$, to agent k consists of all agents that share two or more vertices with agent k.

Definition 3. (Hexagonal lattice formation). The set H of hexagonal lattice formations consists of all agent configurations P such that each agent has exactly 3, 4 or 6 neighbors, all at the inter-agent distance d.

Examples of configurations that do and do not fulfill the definition are given in Figure 1. When considering stability, we use the state vector

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}_0 \\ \vdots \\ \mathbf{p}_9 \end{bmatrix} = \begin{bmatrix} p_{0x} \\ p_{0y} \\ \vdots \\ p_{9x} \\ p_{9y} \end{bmatrix} \in \mathbb{R}^{2K}. \quad (4)$$

The neighborhood of the state $\mathbf{x}$ can be defined as

$$B_\varepsilon(\mathbf{x}) = \{\mathbf{y} : \sqrt{(x_m - y_m)^2 + (x_{m+1} - y_{m+1})^2} < \varepsilon \ \forall m = 0, 2, \ldots, 2(K-1)\}. \quad (5)$$

Also let

$$R(\varphi) = I_K \otimes \begin{bmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi) & \cos(\varphi) \end{bmatrix} \quad (6)$$

and $\mathbf{b} \in \mathbb{R}^2$ be an arbitrary translation (where $\otimes$ denotes the Kronecker product). We can then define the stability we want to achieve.

Definition 4. (Formation stability). The formation $\mathbf{x}$ is stable if

$$\begin{aligned} &\forall \varepsilon > 0 \; \exists \delta > 0 \text{ s.t. } x(0) \in B_\delta(\mathbf{x}) \Rightarrow \\ &\Rightarrow \forall t \; \exists \mathbf{b}, \varphi : x(t) \in B_\varepsilon(R(\varphi)\mathbf{x} + \mathbf{1} \otimes \mathbf{b}). \end{aligned} \quad (7)$$

And it is *asymptotically* stable if it is stable and

$$\begin{aligned} &\exists \delta > 0 \text{ s.t. } x(0) \in B_\delta(\mathbf{x}) \Rightarrow \\ &\Rightarrow \forall t \; \exists \mathbf{b}, \varphi : \lim_{t \to \infty} x(t) = R(\varphi)\mathbf{x} + \mathbf{1} \otimes \mathbf{b}. \end{aligned} \quad (8)$$

Finally we remark that, as described above, if the agents are non-holonomic, we consider the points $\mathbf{p}_k(t)$ as waypoints for a low-level controller. This controller then has the task of driving the agent between waypoints while not leaving the current Voronoi region, R_k. But for the most part of this paper, we will analyze the case of holonomic agents.

3. PROPOSED ALGORITHM

In this section we describe the proposed algorithm in detail and give some remarks on implementational issues. Informally, the algorithm means that each agent computes its Voronoi region and then moves to the centroid of the *vertices* of the region. All vertices are considered as points whose mass is determined by a scalar weight function. To make sure that all Voronoi regions are bounded and thus ensure a balance in the number of vertices, we introduce imaginary neighbors through mirroring of real agents. This mirroring operation is defined as follows.

Definition 5. (Mirror operator). The mirror operator $M : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^2$ is defined as the following mapping

$$M(\mathbf{p}_x, \mathbf{p}_y) = \mathbf{p}_x - \frac{\mathbf{p}_y - \mathbf{p}_x}{||\mathbf{p}_y - \mathbf{p}_x||} \; d. \quad (9)$$

We also adopt the notation $M(k, Q) = \bigcup_{i \in Q} M(\mathbf{p}_k, \mathbf{p}_i)$, i.e. the set of mirror images in agent k of all agents in Q.

Let G_k be the set of perceived neighbors of agent k. By that we mean both real neighbors and mirror images of selected neighbors. Also let $\rho(\mathbf{p}_k)$ be any scalar-valued weight function. The formation control algorithm, that each agent k executes in parallel, can then be described as in Table 1.

Table 1 Formation control algorithm

loop
  Build the neighbor sets N_k and C_k
  if card$(N_k) = 6$ **then**
    $G_k := N_k$
  else if card$(N_k) = 4$ **then**
    $G_k := N_k \bigcup M(k, C_k)$
  else if card$(N_k) = 3$ **then**
    $G_k := N_k \bigcup M(k, N_k)$
  end if
  Compute the centroid of vertices:
  $\mathbf{c} := \left(\sum_{\mathbf{z} \in V_k(G_k)} \rho(\mathbf{z})\right)^{-1} \sum_{\mathbf{z} \in V_k(G_k)} \mathbf{z} \, \rho(\mathbf{z})$
  Apply control: $\mathbf{u}_k := \mathbf{c} - \mathbf{p}_k$
end loop

Remark 1: By assigning different weights $\rho(\mathbf{x})$, we can control the net movement of the formation.

Remark 2: In practice, the agents are likely to have a limited range, $R_{\max}$ of their sensors. But since computing the Voronoi region requires only local knowledge, this does not impose any practical limitations. If a distant agent is beyond sensing range, the Voronoi region may be considered unbounded in that direction. To ensure collision safety, we just limit the control to $|\mathbf{u}| < \frac{1}{2} R_{\max}$. We thus have a decentralized algorithm that can be implemented using only local information.

Remark 3: As formulated above, the algorithm requires that the Voronoi regions are computed synchronously for all agents. In a practical implementation this requirement can be fulfilled by radio synchronization (e.g. using GPS receivers, that give a very accurate time estimate). An alternative solution is to add a safety margin to the Voronoi region borders. If the maximum difference between sampling times for all agents is $\Delta t_{\max}$ and the maximum speed is $v_{\max}$, we can let

$$R_k(Z) := \{\mathbf{x} : |\mathbf{x} - \mathbf{p}_k| < |\mathbf{x} - \mathbf{z}_i| + \Delta t_{\max} \; v_{\max} \; \forall \; i \neq k\}.$$

4. PROPERTIES OF THE ALGORITHM

In this section we show collision safety by using the convexity of the Voronoi regions. We then demonstrate that the hexagonal lattice formation is an asymptotically stable equilibrium. Finally we present numerical arguments that indicate the size of a region of attraction around the equilibrium.

4.1 Safety

Theorem 4.1. (Collision safety). There will be no inter-agent collisions.

Proof We exploit the convexity of the Voronoi regions to show that the centroid for each agent will be in the interior of its Voronoi region, which is disjoint from that of all other agents.

According to (1), $\mathbf{x}$ is inside $R_k(G_k)$ if, for all $i \neq k$

$$
\begin{aligned}
&|\mathbf{p}_k - \mathbf{x}| < |\mathbf{p}_i - \mathbf{x}| \Leftrightarrow (\mathbf{p}_k - \mathbf{x})^2 < (\mathbf{p}_i - \mathbf{x})^2 \Leftrightarrow \\
&\Leftrightarrow 2(\mathbf{p}_i \cdot \mathbf{x} - \mathbf{p}_k \cdot \mathbf{x}) < \mathbf{p}_i \cdot \mathbf{p}_i - \mathbf{p}_k \cdot \mathbf{p}_k \Leftrightarrow \\
&\Leftrightarrow (\mathbf{p}_i - \mathbf{p}_k) \cdot \mathbf{x} < \frac{\mathbf{p}_i + \mathbf{p}_k}{2} \cdot (\mathbf{p}_i - \mathbf{p}_k) \Leftrightarrow \\
&\Leftrightarrow \mathbf{a}^T \mathbf{x} < b, \text{ where} \\
&\mathbf{a} = (\mathbf{p}_i - \mathbf{p}_k) \text{ and } b = \frac{\mathbf{p}_i + \mathbf{p}_k}{2} \cdot \mathbf{a}.
\end{aligned}
$$

Now let $\mathbf{x}_1, \ldots, \mathbf{x}_m$ be points in $R_k(G_k)$. Then any convex combination (e.g. the centroid) of the points is also in the Voronoi region:

$$\sum_n \lambda_n = 1 \Rightarrow \sum_{n=1}^{m} \lambda_n \mathbf{x}_n \in R_k(G_k)$$

since

$$\mathbf{a}^T \sum_{n=1}^{m} \lambda_n \mathbf{x}_n = \sum_{n=1}^{m} \lambda_n \mathbf{a}^T \mathbf{x}_n \leq \sum_{n=1}^{m} \lambda_n b = b \sum_{n=1}^{m} \lambda_n = b.$$

The vertices $V_k(G_k)$ are on the boundary of the Voronoi region, which is not included in it. But except for in degenerate cases, at least one vertex with nonzero weight, V_k^{ij}, is not on a line with all the others. The centroid will then be on the interior of a line between V_k^{ij} and the centroid of all other vertices, which is clearly inside $R_k(G_k)$.

The agent moves along the line between its previous position and the centroid, both of which are inside $R_k(G_k)$. It then follows from the definition of convexity that the new point is inside the region. And since the Voronoi regions of different agents are disjoint by construction, no two agents will ever go to the same point. □

Remark 1: If two agents with limited sensor range are farther than $R_{\max}$ apart, and thus do not sense each other, their Voronoi region estimates may overlap. But if the step size is limited to $\frac{1}{2}R_{\max}$, there cannot be a collision.

Remark 2: The safety property also holds for an agent that does not go straight to the centroid, if it does not leave the Voronoi region or move farther than $\frac{1}{2}R_{\max}$ in one iteration. This motivates the extension of our algorithm to produce waypoints for non-holonomic agents as described in Section 5.

4.2 Stability of formation

We next prove stability for a hexagonal lattice formation of agents. For this analysis we consider a group of 10 agents in a formation depicted in Figure 1b. This formation has been chosen because it is the smallest formation where all types of neighbor sets are represented. Agents have 3, 4 or 6 neighbors. We assume $\rho \equiv 1$, which corresponds to no net movement of the group.

The closed-loop dynamics of the formation is formulated in (10) below. We use the notation $\mathbf{m}(k,i,j)$ for the shared vertex of agents k, i and j, as defined below in (11). If agent i should be mirrored in agent k before the vertex is computed, we denote the vertex as

$$\mathbf{m}(k,\bar{i},j) = \mathbf{m}(k,n,j), \text{ where } \mathbf{p}_n = M(k,\mathbf{p}_i),$$

using the mirror operator defined in (9).

Theorem 4.2. (Local asymptotic stability). The point

$$\mathbf{x}^* = \frac{1}{2}\Big(1,\sqrt{3},3,\sqrt{3},5,\sqrt{3},0,0,2,0,\\ 4,0,6,0,1,-\sqrt{3},3,-\sqrt{3},5,-\sqrt{3}\Big)^T$$

is a locally asymptotically stable formation for the system (10).

Proof We linearize the system (10) around the stationary point $\mathbf{x}^*$ and show that the eigenvalues of the Jacobian are all on or inside the unit circle. Finally we note that all eigenvectors corresponding to an eigenvalue on the unit circle describe rotations of the whole formation with preserved relative positions of the agents.

Using computer software for symbolical algebra, we find that the Jacobian

$$J = \left.\frac{d\mathbf{f}}{d\mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^*} \tag{12}$$

has two unit eigenvalues and that all others are inside the unit circle. All eigenvalues inside the unit circle correspond to perturbations that the algorithm will attenuate. The eigenvectors with unit eigenvalues correspond to rotations of the whole formation (clockwise and counterclockwise). Under these perturbations, all relative positions of the agents are preserved and, according to (8), this does not affect the asymptotic stability of the formation. □

4.3 Region of attraction

To further test the stability properties of our proposed algorithm, we investigate what happens if we perturb the position of one single agent in a hexagonal lattice where all other agents are fix. We then let the agent move according to one iteration of the algorithm and plot the direction of its movement as a function of the perturbation. This yields a vector field that indicates what magnitude of perturbations the algorithm can handle.

We only study one iteration since in a real situation, where all agents move, in the next iteration all surrounding agents will have moved too, so the simplification does not hold anymore. But if the perturbed agent has then taken a step towards the origin, it is plausible that the whole formation converges (which is also confirmed by simulations in Section 5).

$$\mathbf{x}(t+1) = \mathbf{f}(\mathbf{x}(t)) = \frac{1}{6}\begin{bmatrix} \mathbf{m}(0,1,4)+\mathbf{m}(0,3,4)+\mathbf{m}(0,1,\bar{3})+\mathbf{m}(0,3,\bar{1})+\mathbf{m}(0,\bar{1},\bar{4})+\mathbf{m}(0,\bar{4},\bar{3}) \\ \mathbf{m}(1,0,4)+\mathbf{m}(1,4,5)+\mathbf{m}(1,2,\underline{5})+\mathbf{m}(1,\underline{0},5)+\mathbf{m}(1,\underline{2},\underline{4})+\mathbf{m}(1,\underline{4},\underline{5}) \\ \mathbf{m}(2,1,5)+\mathbf{m}(2,5,6)+\mathbf{m}(2,1,\underline{6})+\mathbf{m}(2,\underline{1},6)+\mathbf{m}(2,\underline{5},\underline{6})+\mathbf{m}(2,\underline{1},\underline{5}) \\ \mathbf{m}(3,0,4)+\mathbf{m}(3,4,7)+\mathbf{m}(3,0,\bar{7})+\mathbf{m}(3,0,7)+\mathbf{m}(3,\underline{0},\underline{4})+\mathbf{m}(3,\underline{4},\underline{7}) \\ \mathbf{m}(4,0,1)+\mathbf{m}(4,1,5)+\mathbf{m}(4,5,8)+\mathbf{m}(4,7,8)+\mathbf{m}(4,\underline{3},\underline{7})+\mathbf{m}(4,\underline{0},\underline{3}) \\ \mathbf{m}(5,1,2)+\mathbf{m}(5,2,6)+\mathbf{m}(5,6,\underline{9})+\mathbf{m}(5,\underline{8},\underline{9})+\mathbf{m}(5,\underline{4},\underline{8})+\mathbf{m}(5,\underline{1},\underline{4}) \\ \mathbf{m}(6,2,5)+\mathbf{m}(6,5,9)+\mathbf{m}(6,\underline{2},\underline{9})+\mathbf{m}(6,\underline{2},\underline{5})+\mathbf{m}(6,\underline{5},\underline{9}) \\ \mathbf{m}(7,3,4)+\mathbf{m}(7,4,8)+\mathbf{m}(7,\underline{3},\underline{8})+\mathbf{m}(7,\underline{3},\underline{4},)+\mathbf{m}(7,\underline{4},\underline{8}) \\ \mathbf{m}(8,4,7)+\mathbf{m}(8,4,5)+\mathbf{m}(8,\underline{5},\underline{9})+\mathbf{m}(8,\underline{4},\underline{9})+\mathbf{m}(8,\underline{5},\underline{7})+\mathbf{m}(8,\underline{4},\underline{5}) \\ \mathbf{m}(9,5,8)+\mathbf{m}(9,5,6)+\mathbf{m}(9,6,8)+\mathbf{m}(9,6,8)+\mathbf{m}(9,\underline{5},\underline{8})+\mathbf{m}(9,\underline{5},\underline{6}) \end{bmatrix} \tag{10}$$

$$\mathbf{m}(k,i,j) = \begin{bmatrix} \frac{1}{2}\dfrac{(p_{jy}-p_{ky})\left[p_{ix}^2+p_{iy}^2-(p_{kx}^2+p_{ky}^2)\right]+(p_{ky}-p_{iy})\left[p_{jx}^2+p_{jy}^2-(p_{kx}^2+p_{ky}^2)\right]}{(p_{ix}-p_{kx})(p_{jy}-p_{ky})-(p_{iy}-p_{ky})(p_{jx}-p_{kx})} \\ \frac{1}{2}\dfrac{(p_{kx}-p_{jx})\left[p_{ix}^2+p_{iy}^2-(p_{kx}^2+p_{ky}^2)\right]+(p_{ix}-p_{kx})\left[p_{jx}^2+p_{jy}^2-(p_{kx}^2+p_{ky}^2)\right]}{(p_{ix}-p_{kx})(p_{jy}-p_{ky})-(p_{iy}-p_{ky})(p_{jx}-p_{kx})} \end{bmatrix} \tag{11}$$

To simplify the calculations, we translate the agent to the origin, surrounded by six neighbors:

$$\begin{aligned} \mathbf{p}_0 = -\mathbf{p}_3 &= (1,0)^T \\ \mathbf{p}_1 = -\mathbf{p}_4 &= \frac{1}{2}(1,\sqrt{3})^T \\ \mathbf{p}_2 = -\mathbf{p}_5 &= \frac{1}{2}(-1,\sqrt{3})^T \end{aligned}$$

The simplification of considering only six neighbors holds for perturbations of magnitude less than $1/\sqrt{3}$. This is because the Voronoi region of the center agent will not be affected by any of the other agents in the lattice. We also assume $\rho \equiv 1$.

If the center agent is perturbed to the position $\mathbf{r} = (r_x\ r_y)^T$, after one iteration of the algorithm, it will be at $\mathbf{c} = (c_x\ c_y)^T$. The direction of the step $\mathbf{r}-\mathbf{x}$ as a function of $\mathbf{x}$ can be plotted as a vector field, depicted in Figure 2. A circle with radius $1/\sqrt{3}$ shows the region where the assumption of six neighbors is valid. Within this circle, the vector field indicates that the algorithm should be convergent. So a reasonable estimate of the region of attraction appears to be a circle of radius $1/\sqrt{3}$ centered around the perturbed agent, something that is also verified by simulations in the following section. This also means that one single agent may only be displaced by this much from an ideal hexagonal lattice formation when the algorithm is initiated. Otherwise the group may not converge to the desired formation.

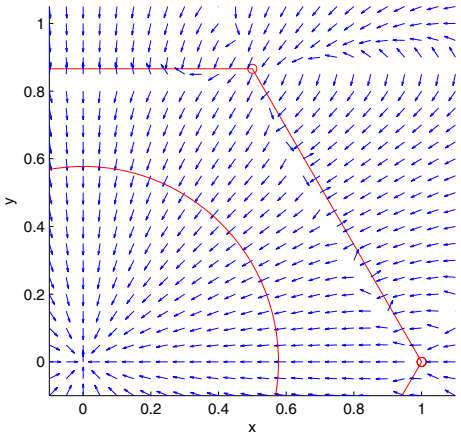

Fig. 2. A vector field with normalized vector lengths, showing the direction of $\mathbf{r}-\mathbf{x}$ as a function of $\mathbf{x}$. The circle shows the region where only the six closest neighbors affect the Voronoi region of

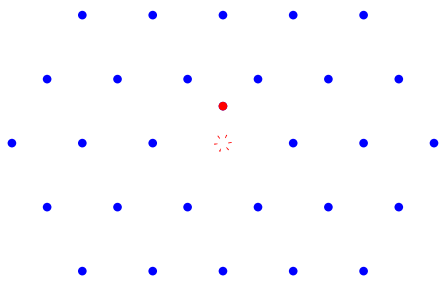

Fig. 3. A formation where the middle agent has been displaced the distance $0.5d$ in the most sensitive direction. The dashed circle shows its original position. This perturbation is attenuated in a few iteration steps.

5. SIMULATIONS

We simulate a larger group of agents that start in a hexagonal lattice formation and all evolve according to the algorithm. Three scenarios are studied: First one single agent in a stationary formation is perturbed, to test stability. Then we move the whole formation while adding noise to the control signal to each agent, to simulate the effects of uneven terrain and platform imperfections. Finally we apply the algorithm to a group of car-like robots, to illustrate the usefulness of the hierarchical controller architecture.

By perturbing one of the agents in a stationary formation (where the weight ρ is constant), we can study the asymptotic stability. Figure 3 depicts such a formation where the middle agent is perturbed by $0.5d$, which is attenuated in a few iteration steps. The original position of the agent is shown by a dashed circle. The maximum perturbation that can be attenuated even in the most sensitive direction is found to be $0.57d \approx 1/\sqrt{3}$. This concurs well with the results in Section 3.

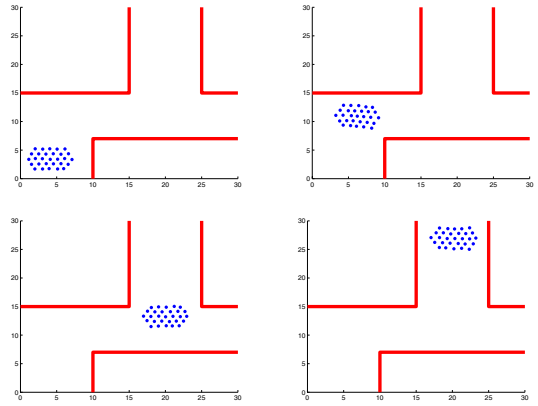

Fig. 4. Snapshots of a group of 29 agents moving according to a linear weight function whose slope is changed to control the direction of movements. The whole sequence takes 160 iterations.

In the second simulation we control the global movement of a formation by changing the slope of the weight function. We use the weight

$$\rho(x,y) = e^{x\cos(\varphi)+y\sin(\varphi)}, \tag{13}$$

where we can vary the angle φ over time to make the group move in different directions. We have chosen the exponential function to get uniform speed of movement over the whole plane. The result in Figure 4 shows snapshots of the formation being steered around obstacles. We add noise to the control signal of each agent to simulate uneven terrain and other errors:

$$\mathbf{u}_k = \mathbf{c} - \mathbf{p}_k + \begin{bmatrix} \theta_x \\ \theta_y \end{bmatrix} \tag{14}$$

The random variables θ_x, θ_y are evenly distributed in the interval $[0, 0.15]$.

Finally we study a more realistic model, a kinematic car (Murray and Sastry, 1993), with the forward velocity v and steering angle δ as controls:

$$\begin{cases} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= \dfrac{v}{L}\tan\delta \end{cases} \tag{15}$$

Each car has a controller whose task is to drive the car between waypoints, while respecting the safe regions designated by our higher-level algorithm. This hierarchy is illustrated in Figure 5.

A formation of 13 kinematic cars is steered using the weight (13) in a 100×100 m^2 labyrinth, with the inter-agent distance d=5 m. Since each car must not leave its Voronoi region, they must sometimes perform parallel parking-like maneuvers to turn towards the next waypoint produced by our algorithm. Figure 6 shows snapshots of the group at four different time instances. Here, too, we add noise (14) to the waypoints given to the lower-level car controllers. With a maximum car speed of 1 m/s, the whole trajectory is completed in about 2.5 minutes. These simulations illustrate the validity of the stability analysis and the feasibility of

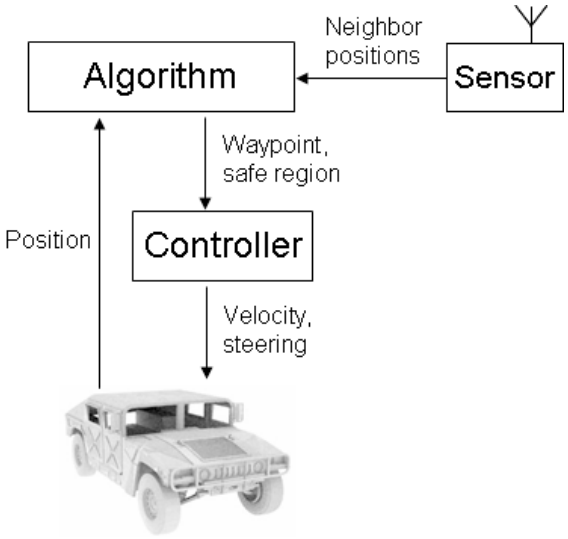

Fig. 5. The hierarchical architecture where our algorithm for holonomic agents feeds a waypoint and a corresponding safe region to a lower-level controller that drives the non-holonomic car.

structuring the control of non-holonomic agents in two levels, where the top level algorithm produces waypoints and associated safe regions for maneuvering.

6. CONCLUSIONS

We have presented an algorithm that offers provable safety from collisions, is scalable with group size and requires a minimum of inter-agent communications. The only information needed by every agent is the relative positions of its neighbors. The algorithm not only computes waypoints, but also associates a safe region to every waypoint, that a controller for a non-holonomic agent can use for maneuvering. It also allows the incorporation of a scalar weight as a means of controlling the net movement of the group. We have proved local asymptotic stability and shown simulations that indicate the magnitude of allowed perturbations. Finally we have illustrated the possibility to use the algorithm in a hierarchical control architecture, with kinematic car agents.

In future work we hope to be able to formally show convergence of the whole formation and also devise a mirroring strategy that allows dynamically changing Voronoi neighbor graphs. This could enable us to start in any constellation and converge to the hexagonal lattice formation.

We also plan to eventually implement the algorithm on the same simulation platform where we have tested our previous work (Lindhé *et al.*, 2005). This is a very detailed physics engine where we have modelled each agent as a US Army HMMWV jeep, Figure (7). The controller of each car controls the steering angle, accelerator and brakes and has access to the positions of all other cars by shared variables. Including many real-world effects such as wheel slip and uneven terrain,

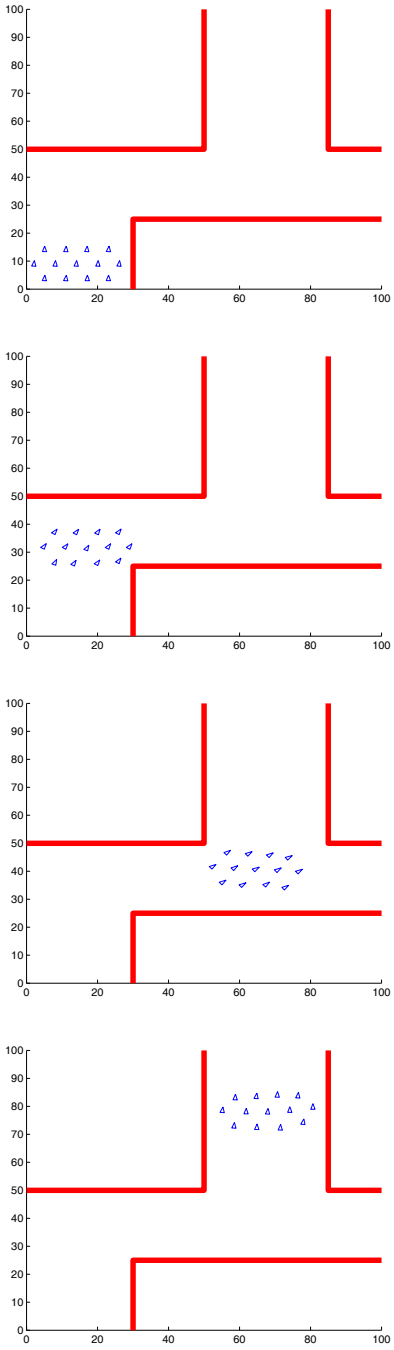

Fig. 6. Snapshots of a formation of 13 robots with car-like dynamics moving through a simple labyrinth. The labyrinth is square, with side length 100 m.

we believe that this simulator environment is a very realistic setting for testing of our algorithm.

Fig. 7. The US Army HMMWV jeep model that we plan to use as agents in our simulations. In the background we see trenches used as obstacles.

REFERENCES

Beard, R.W., J. Lawton and F. Y. Hadaegh (2001). A Coordination Architecture for Spacecraft Formation Control. *IEEE Transactions on Control Systems Technology*.

Cortés, J., S. Martínez, T. Karataş and F. Bullo (2004). Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation* **20**(2), 243–255.

Desai, J. P., J. Ostrowski and V. Kumar (1998). Controlling formations of multiple mobile robots. In: *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*. pp. 2864–2869. Leuven, Belgium, May 1998.

Healey, A.J. (2001). Application of formation control for multi-vehicle robotic minesweeping. In: *Proceedings of the 40th IEEE Conference on Decision and Control*. Vol. 2. pp. 1497–1502.

Leonard, N. E. and E. Fiorelli (2001). Virtual Leaders, Artificial Potentials and Coordinated Control of Groups. In: *Proceedings of the 40th IEEE Conference on Decision and Control*. pp. 2968–2973.

Lindhé, M., P. Ögren and K. H. Johansson (2005). Flocking with Obstacle Avoidance: A New Distributed Coordination Algorithm Based on Voronoi Partitions. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. pp. 1785–1790.

Liu, Y., K.M. Passino and M.M. Polycarpou (2003). Stability analysis of M-dimensional asynchronous swarms with a fixed communication topology. *IEEE Transactions on Automatic Control* **48**, 76–95.

Murray, R. M. and S. S. Sastry (1993). Nonholonomic motion planning: Steering using sinusoids. *IEEE Transactions on Automatic Control* **38**(5), 700–716.

Nguyen, H. G., N. Farrington and N. Pezeshkian (2004). Maintaining Communication Link for Tactical Ground Robots. In: *AUVSI Unmanned Systems North America 2005*.

Reynolds, C. (1987). Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics*.

Scharf, D. P., F. Y. Hadaegh and S. R. Ploen (2004). A Survey of Spaceraft Formation Flying Guidance and Control (Part II): Control. In: *Proceeding of the 2004 American Control Conference*.

Tanner, H. G., A. Jadbabaie and G. J. Pappas (To appear). Flocking in Fixed and Switching Networks. *IEEE Transactions on Automatic Control*.

Trianni, V., S. Nolfi and M. Dorigo (2005). Cooperative Hole Avoidance in a *Swarm-bot*. *Robotics and Autonomous Systems*. to appear.