Proceedings of the ACRA 2005,
Sydney, 5-7 December

THE AUSTRALIAN NATIONAL UNIVERSITY

Control of Contour Formations of Autonomous Vehicles by General Curve Evolution Theory

Shahab Kalantar & Uwe R. Zimmer

Australian National University
Research School for Information Science and Engineering
and the Faculty for Engineering and Information Technology[1]
Autonomous Underwater Robotics Research Group
Canberra, ACT 0200, Australia
shahab.kalantar@rsise.anu.edu.au | uwe.zimmer@ieee.org

In this paper, we introduce the concept of a contour-shaped formation of autonomous underwater vehicles for the purpose of adaptation to isoclines of environmental fields created as a result of diffusion of chemicals. General curvature-driven curve evolution theory is used as a continuum model for such formations. A collection of robots can act as markers on an imaginary curve moving according to the local curvature and the external environmental force projected unto the normal to the curve. Basic control rules are given, and various strategies for implementation of the scheme are explored. Algorithms are discussed for formation initialization, and simulation runs are presented to demonstrate the behaviour of these types of formations.

1. Introduction

Using a group of mobile robots to localize, track, converge to, or enclose the source of a *plume* released into an environment has gained much attention in robotics communities. Often, the robots have to form special shapes dictated by the application. In this paper, we look at curve-like groups which are required to seek and remain on a particular contour around the plume source. A *formation* Φ of a set of n mobile robots R_i, with states $q_i(t) \in \Re^s$, can be defined as a collection of geometric relationships constraining the aggregate state $q \in \Re^{sn}$. A *formation function* $\mathcal{V}_\Phi(q)$ can be designed to capture these constraints. In a *rigid* formation, we only have strict equalities and $\mathcal{V}_\Phi : \Re^{sn} \to \Re$ (a scalar function) can be defined in such a way that a perfect formation would satisfy $\mathcal{V}_\Phi(q) = 0$. The only degrees of freedom are those of translation and rotation. With *deformable* (non-rigid) formations, we have much more degrees of freedom. In this case, the vector-valued $\vec{\mathcal{V}}_\Phi : \Re^{sn} \to \Re^r$ (r is the number of constraints) would constitute a number of inequalities as well and a vector inequality such as $\vec{\mathcal{V}}_\Phi(q) \le \vec{0}$ may describe an infinite number of acceptable formations (even when translation and rotation have been abstracted away). *Formation control* should keep the aggregate state on the *formation manifold*. Formations are also required to fulfil some sort of *mission*. We can represent a mission by a suitable mission function $\mathcal{M}_\Phi : \Re^{sn} \otimes \Psi \to \Re$, where Ψ is a set of parameters. Mission is accomplished if $\mathcal{M}_\Phi = 0$.

Rigid formations have been the topic of great many research papers (see, e.g., [3, 7, 8]), while deformable ones have not been addressed sufficiently (see [5, 9] for an example of research on deformable formations). In our particular application, a contour-shaped array of robots should adapt itself to a contour $\gamma_{\mathcal{F}_d}$, corresponding to a desired isocline (or level sets) $\mathcal{F}_d$ of a distribution. The contour formation should satisfy constraints related to *bending* and *stretching* of the band, so that $\mathcal{V}_\Phi$ is composed of a set of inequalities such as $d_1 \le \|q_i - q_{i-1}\| \le d_2$, $\|q_i - q_j\| \le d_3$, $|\kappa_i| \le \kappa_0$, where it is assumed that the indexes define a total ordering between robots. κ denotes curvature. Furthermore, $\mathcal{M}_\Phi$ is defined as

$$\sum_{i=1}^{n} \left\| Q(\gamma_{\mathcal{F}_d}, q_i) - q_i \right\|, \qquad (1)$$

where $Q(\gamma_{\mathcal{F}_d}, \chi)$ denotes the closest point on the level set to point χ. As $t \to \infty$, it is desired that $\mathcal{M}_\Phi \to 0$, while $\vec{\mathcal{V}}_\Phi(q) \le \vec{0}$ at all times. The approach we take in this paper is inspired by similar applications of deformable models in machine vision.

Environmental isoclines can be defined as contours of equal concentration created as a result of *diffusion* of some chemical released from a source. In its simplest form, diffusion follows Fick's law, which states

[1] *Part of this research was done under a grant from Cooperative Research Centre for Intelligent Manufacturing Systems and Technology (CRC-IMST), Australia.*

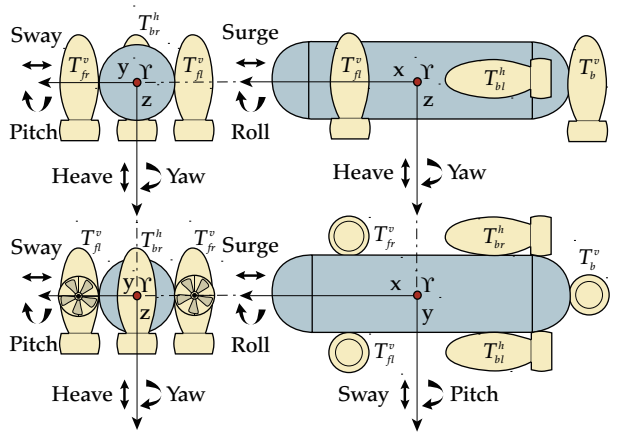

figure 1: Serafina underwater vehicle

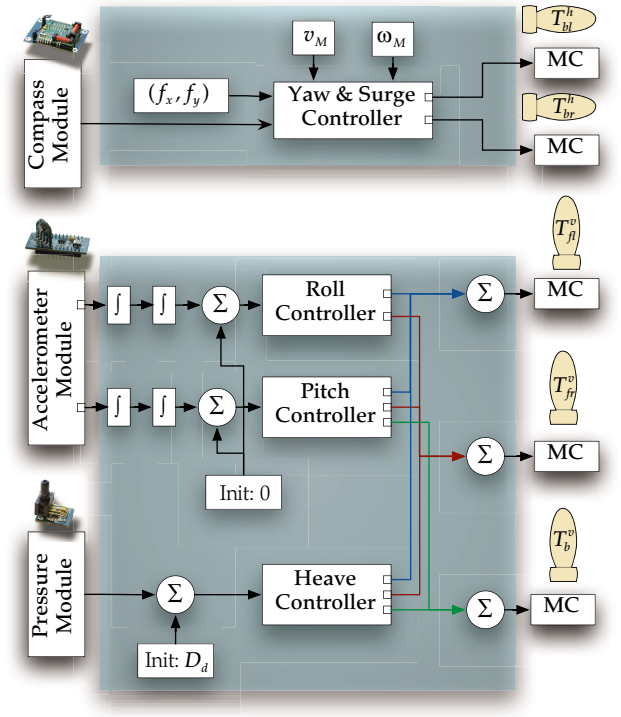

figure 2: Decoupled control system for Serafina. $[f_x, f_y]$ is the target position measured with respect to the body-fixed coordinate Υ. v_M and ω_M are bounds on linear and angular velocities, respectively.

that the flux of matter is a function of the gradient of the concentration of matter, i.e.,

$$J_q = -D_x \frac{\partial C(q)}{\partial x} - D_y \frac{\partial C(q)}{\partial y} \tag{2}$$

and the movement of particles of matter follows

$$\frac{\partial C(q)}{\partial t} = -\frac{J_q}{\partial q} \tag{3}$$

where $C(q)$ denotes the concentration at $q \in \Re^2$, D_x and D_y are coefficients of diffusion. A simpler and more realistic method for modelling plumes is to use the *similarity hypothesis* ([2]), which states that, as the patch is spread, the shape of the particle distribution is preserved. Thus, the mean concentration distribution would be

$$\bar{C}(q,t) = \frac{Q(t)}{\beta \sigma_x(t)\sigma_y(t)} G\left\{\frac{x}{\sigma_x(t)}, \frac{y}{\sigma_y(t)}\right\} \tag{4}$$

where $Q(t)$ is the rate of release of particles, and $\sigma_x(t)$ and $\sigma_y(t)$ are, respectively, the standard deviations in the respective directions, and implicitly describe environmental conditions. the function G determines the shape of the distribution, and β is a constant depending on G. Usually, a Gaussian distribution for G is assumed. In this paper, we will deal with slow diffusion processes, i.e., it is assumed that adaptation occurs much faster than the spread of the plume. Alternatively, it may be assumed that the rate of release is bigger than the rate of change of the standard deviations. If this is not the case, then, after reaching steady state, the plume will form a uniform patch. For the particular case of *salinity* or *temperature*, which we are primarily interested in, this condition is satisfied. Furthermore, we assume that the flow is almost *laminar*, so that we can ignore the effect of *turbulence*. Finally, external forces (wind, ocean flow) always affect the spatial distribution. They may counteract or augment diffusion. As will be shown later, the only information the robots need to ascent or descend the distribution towards the desired level set, is the local gradients of concentration. The minimal requirement is that this gradient be reasonably well-defined and smooth. Thus, the geometric form of the plume is irrelevant, as long as its scale is commensurate with the size of the formation. See [2] for more detailed information on plumes and environmental diffusion.

The control scheme to be presented in this paper is intended to be implemented on small submarines called Serafina, Developed at our laboratory. Figure 1 shows this vehicle from different views. We confine ourselves to motion on the plane. As shown in figure 2, *roll and pitch controllers* try to neutralize the effect of disturbances on roll and pitch angles. The *heave controller* keeps the robot at a certain desired depth D_d, relative to the ocean bottom. Motion of the submarine on the x-y plane is thus similar to a *non-holonomic* vehicle because swaying is not directly controllable. The *yaw and surge controller* guides the robot on the plane to desired locations. See [10] for details on dynamic modelling and decoupled control of marine vehicles.

2. Deformable contour formations

Before discussing actual formations, let us study the case of an idealized smooth curve which can be regarded as a *continuum model* for contour formations. For more details and arguments, see [4] and references there in. Let $\gamma\colon [0,1]\otimes\Re\to\Re^2$ define a moving curve on the plane, parametrized by arc-length s. Thus $\gamma(s,t)=[y(s,t),x(s,t)]^T$ gives the position in $\Re^2$ of points constituting the curve. The general curve evolution theory is based on the fact that the flow

$$\frac{\partial\gamma(s,t)}{\partial t}=\kappa(s,t)\vec{N}(s,t) \tag{5}$$

moves the curve in the direction of the gradient of $L(\gamma)=\oint ds$, i.e., reduces the length as fast as possible and is called the Euclidian *curve shortening flow*, where

$$\kappa(s,t)=\frac{y_{ss}(t)x_s(t)-x_{ss}(t)y_s(t)}{\sqrt{(x_s(t)^2+y_s(t)^2)^3}} \tag{6}$$

is the *Euclidian curvature* and

$$\vec{N}(s,t)=\frac{1}{\sqrt{x_s(t)^2+y_s(t)^2}}\begin{pmatrix}y_s(t)\\-x_s(t)\end{pmatrix} \tag{7}$$

is the unit *inward normal* to the curve at $\gamma(s,t)$. Adaptation of γ to a desired level set $\mathcal{F}_d$ of the diffused plume $\mathcal{F}(q)$ can be formulated as the minimization of the functional

$$\mathcal{E}(\gamma)=\int_0^{L(\gamma)}g(\gamma(s))ds \tag{8}$$

where $g\colon[0,\max_q\mathcal{F}]\to\Re^+$ is a strictly decreasing function, i.e, $g(q)\to 0$ as $\mathcal{F}(q)\to\mathcal{F}_d$. g is related to $\mathcal{F}$ through the definition $g(q)=h_{\mathcal{F}_d}\circ\mathcal{F}(q)$. $h_{\mathcal{F}_d}$ serves to stop the curve from evolving past the desired iso-cline and a good choice for it is

$$h_{\mathcal{F}_d}(\zeta)=1-\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{|\zeta-\mathcal{F}_d|^2}{\sigma^2}} \tag{9}$$

The gradient descent of $\mathcal{E}(\gamma)$ gives the flow

$$\begin{aligned}\partial\gamma(s,t)/\partial t&=g(\gamma(s,t))\kappa(s,t)\vec{N}(s,t)\\&-\langle\nabla_{\gamma(s,t)}g(\gamma(s,t)),\vec{N}(s,t)\rangle(\vec{N}(s,t))\end{aligned} \tag{10}$$

where

$$\nabla_q g(q)=\nabla_{\mathcal{F}(q)}h_{\mathcal{F}}(\mathcal{F}(q))\nabla_q\mathcal{F}(q) \tag{11}$$

In the discrete implementation of the contour evolution, where the curve is composed of a finite number of nodes (as is the case in a robotic contour formation), moving towards the normal to the curve will cause the nodes to come very close to each other. On the other hand, in an open contour, it would be desirable to keep the length of the curve fixed, which is not enforced by normal motion alone. To remedy these problems, some sort of attractive-repulsive forces are required. It can be proved that if these forces act towards the tangent to the curve, the geometry of the curve will not be affected. It will only change the parametrization in the continuum model. This will lead to a more general form of curve evolution

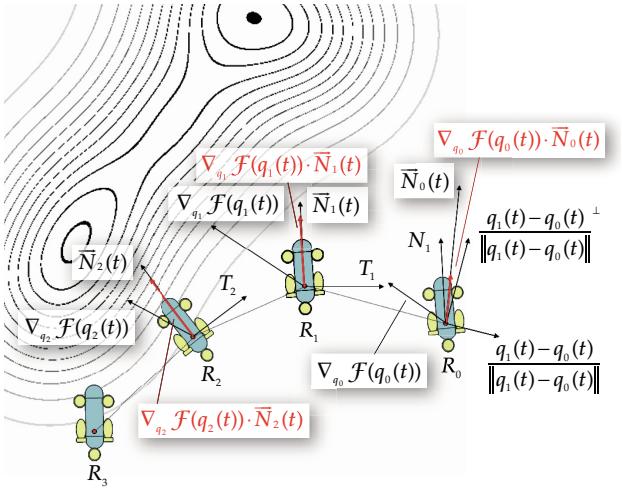

figure 3: Contour formation

$$\frac{\gamma(s,t)}{\partial t}=\eta_N(s,t)\vec{N}(s,t)+\eta_T(s,t)\vec{T}(s,t) \tag{12}$$

where

$$\vec{T}(s,t)=\frac{\partial\gamma(s,t)/\partial s}{\|\partial\gamma(s,t)/\partial s\|}=\vec{T}(s,t)^{\perp} \tag{13}$$

is the tangent to the curve at $\gamma(s,t)$. The non-zero *tangential component* η_T can be designed according to desired behaviour. For a closed curve, we use the boundary condition $\gamma(1,t)=\gamma(0,t)$. For open curves, the end points are kept fixed in time $\dot{\gamma}(0,t)=\dot{\gamma}(1,t)=0$.

Based on the above continuum model for curve motion under the influence of an external force, we now proceed to define control laws for the motion of robots R_i. Let us set

$$\frac{\partial q_i(t)}{\partial t}=\eta_{N_i}(t)\vec{N}_i(t)+\eta_{T_i}(t)\vec{T}_i(t) \tag{14}$$

where

$$\begin{aligned}\eta_{N_i}(t)&=\beta_1\kappa_i(t)g(q_i(t))\\&-\beta_2\langle\dot{h}_{\mathcal{F}}(\mathcal{F}(q_i(t)))\nabla_{q_i(t)}\mathcal{F}(q_i(t)),\vec{N}_i(t)\rangle\end{aligned} \tag{15}$$

The normal, and tangent vectors, and the curvature are defined, respectively, by

$$\vec{N}_i(t)=\frac{\begin{pmatrix}(y_{i+1}(t)-y_{i-1}(t))\\-(x_{i+1}(t)-x_{i-1}(t))\end{pmatrix}}{\|q_{i+1}(t)-q_{i-1}(t)\|}$$

$$\vec{T}_i(t)=\frac{q_{i+1}(t)-q_{i-1}(t)}{\|q_{i+1}(t)-q_{i-1}(t)\|}=\vec{N}_i(t)^{\perp}$$

$$\kappa_i(t)=\frac{4}{\|q_{i+1}(t)-q_{i-1}(t)\|^3}\tilde{\kappa}(t)$$

$$\tilde{\kappa}(t) = (y_{i-1}(t) - 2y_i(t) + y_{i+1}(t))(x_{i+1}(t) - x_{i-1}(t)) - (x_{i-1}(t) - 2x_i(t) + x_{i+1}(t))(y_{i+1}(t) - y_{i-1}(t)) \quad (16)$$

To maintain a rigid inter-robot distance $d = L/N$, where L is a desired contour length, we define

$$\eta_{T_i}(t) = \beta_3(e^{((\|q_i(t)-q_{i-1}(t)\|-d)/p_i(t))^2} - e^{((\|q_{i+1}(t)-q_i(t)\|-d)/p_i(t))^2}) \quad (17)$$

where

$$p_i(t) = \|q_i(t) - q_{i-1}(t)\| + \|q_{i+1}(t) - q_i(t)\| \quad (18)$$

The end robots, in an open contour formation, have to be treated differently because the curvature is not defined at these two positions. Moreover, the normal direction is ill-defined. In [2], we let the two end robots move according to the following rules:

$$\dot{q}_0(t) = -\beta_4 g(q_0(t))\nabla_{q_0(t)}\mathcal{F}(q_0(t)) - \beta_5 e^{\left(\frac{(\|q_1(t)-q_0(t)\|-d)}{\|q_1(t)-q_0(t)\|}\right)^2} \frac{\nabla^{\perp}_{q_0(t)}\mathcal{F}(q_0(t))}{\left\|\nabla^{\perp}_{q_0(t)}\mathcal{F}(q_0(t))\right\|}$$

$$\dot{q}_{N-1}(t) = -\beta_4 g(q_{N-1}(t))\nabla_{q_{N-1}(t)}\mathcal{F}(q_{N-1}(t)) - \beta_5 e^{\left(\frac{(\|q_{N-1}(t)-q_{N-2}(t)\|-d)}{\|q_{N-1}(t)-q_{N-2}(t)\|}\right)^2} \frac{\nabla^{\perp}_{q_{N-1}(t)}\mathcal{F}(q_{N-1}(t))}{\left\|\nabla^{\perp}_{q_{N-1}(t)}\mathcal{F}(q_{N-1}(t))\right\|} \quad (19)$$

These work fine if the smaller angle between $\nabla_{q_0(t)}\mathcal{F}(q_0(t))$ and $q_1(t) - q_0(t)$, and similarly, the angle between $\nabla_{q_{N-1}(t)}\mathcal{F}(q_{N-1}(t))$ and $q_{N-1}(t) - q_{N-2}(t)$, are sufficiently large. If this is not the case, then the direction of the gradient may be almost co-linear with the local tangent to the curve, so that the motion of either or both of the end robots can make the contour to shrink or stretch. To remedy this problem, we propose a different strategy, at the expense of some extra information. We define the normal at either of the end points as the mean of the normal at the adjacent robot and the unit vector perpendicular to the vector joining two adjacent robots:

$$\dot{q}_o(t) = -\beta_4 g(q_0(t)) \langle \dot{h}_{\mathcal{F}}(\mathcal{F}(q_0(t)))\nabla_{q_0(t)}\mathcal{F}(q_0(t)), \vec{N}_0(t)\rangle + \frac{1}{2}\beta_5 e^{\left(\frac{\|q_1(t)-q_0(t)\|-d}{\|q_1(t)-q_0(t)\|}\right)^2}\left(\vec{T}_1(t) + \frac{q_1(t)-q_0(t)}{\|q_1(t)-q_0(t)\|}\right) \quad (20)$$

$$\vec{N}_0(t) = \frac{1}{2}\left(\vec{N}_1(t) + \frac{q_1(t)-q_0(t)}{\|q_1(t)-q_0(t)\|}^{\perp}\right) \quad (21)$$

$$\dot{q}_{N-1}(t) = -\beta_4 g(q_{N-1}(t)) \langle \dot{h}_{\mathcal{F}}(\mathcal{F}(q_{N-1}(t)))\nabla_{q_{N-1}(t)}\mathcal{F}(q_{N-1}(t)), \vec{N}_{N-1}(t)\rangle + \frac{1}{2}\beta_5 e^{\left(\frac{\|q_{N-1}(t)-q_{N-2}(t)\|-d}{\|q_{N-1}(t)-q_{N-2}(t)\|}\right)^2} \left(\vec{T}_{N-2}(t) + \frac{q_{N-2}(t)-q_{N-1}(t)}{\|q_{N-2}(t)-q_{N-1}(t)\|}\right) \quad (22)$$

$$\vec{N}_{N-1}(t) = \frac{1}{2}\left(\vec{N}_{N-2}(t) + \frac{q_{N-2}(t)-q_{N-1}(t)}{\|q_{N-2}(t)-q_{N-1}(t)\|}^{\perp}\right) \quad (23)$$

Note that this may hinder the formation from adapting but it will ensure its integrity. See figure 3 for a visual demonstration of vectors. This particular choice may be problematic with very large curvatures. On the other hand, it may help the end robots escape stagnation when $\nabla_{q_0(t)}\mathcal{F}(q_0(t))$ becomes almost co-linear with T_0 (and similar for q_N).

3. Implementation and hybrid modelling

When implementing the control scheme described in the previous section, several possibilities can be considered. We have assumed that each robot can measure the gradient at its position so that the only external information needed by each robot are the positions of its two neighboring robots with respect to its body-fixed local coordinate system. This information can be gained through *active sensing*, *explicit communication*, or, more realistically, a combination of both. There is an inevitable delay associated with this process. Figure 4.a shows a simple model for communication channels between adjacent robots. In a *fully synchronized* scheme, the robot formation implements a discrete version of equation (14): at each discrete instant t, all the robots start synchronizing on shared information, while keeping station, before moving to the next location given by

$$q^d_{t+\Delta t} = q(t) + \Delta t(\eta_N(t)N(t) + \eta_T(t)T(t)) \quad (24)$$

using the linear control rule

$$\dot{q}(\tau) = \nu\frac{q^d_{t+\Delta t} - q(t)}{\left\|q^d_{t+\Delta t} - q(t)\right\|} \quad (25)$$

where ν ensures that $q^d_{t+\Delta t}$ is reached in Δt. In practice, synchronization takes some time, so that there will be intervals of inaction in between successive movements. Δt is selected based on the characteristics of the differential equation ([9]) and should, in general, be small enough. Figure 4.b shows the timed automaton for the synchronization phase. In a *fully asynchronous* scheme, we have

$$\dot{q}_i(t) = \Xi(q_i(t), q_{i-1}(t-\tau_-(t)), q_{i+1}(t-\tau_+(t))) \quad (26)$$

where Ξ denotes the curve evolution equation, and $\tau_-(t)$ and $\tau_+(t)$ denote, respectively, the times (measured from t) when the positions q_{i-1} and q_{i+1} were updated. If $|\tau_+(t) - \tau_-(t)|$, $t - \tau_-(t)$, and $t - \tau_+(t)$ are within reasonable bounds, equation (26) would be nearly as good as equation (24). The bounds should be determined through analysis and, in general, depend on the speed of individual robots. If the bounds can become relatively large, we can extrapolate the motions of R_{i-1} and R_{i+1} using the laws

$$\begin{aligned}\hat{q}_{i-1}(t) &= q_{i-1}(t-\tau_-) + (t-\tau_-)\hat{v}_{i-1}(t-\tau_-)\\ \hat{q}_{i+1}(t) &= q_{i+1}(t-\tau_+) + (t-\tau_+)\hat{v}_{i+1}(t-\tau_+)\end{aligned} \quad (27)$$

where $\hat{v}_{i+1}$ and $\hat{v}_{i-1}$ denote the estimated velocities at the time when q_{i+1} and q_{i-1} were last updated. It is desirable to bound

$$\begin{aligned}\|q_i(t) - \hat{q}_i(t)\| &= \left\| \int_{t_0}^{t} [\dot{q}_i(t) - \dot{\hat{q}}_i(t)]dt \right\| \quad (28)\\ &\le \int_{t_0}^{t} [\dot{q}_i(t) - \dot{\hat{q}}_i(t)]dt\\ &\le \int_{t_0}^{t} (\left\| \eta_{N_i} \vec{N}_i(t) - \tilde{\eta}_{N_i} \vec{\tilde{N}}_i(t) \right\|\\ &\quad + \left\| \eta_{T_i} \vec{T}_i(t) - \tilde{\eta}_{T_i} \vec{\tilde{T}}_i(t) \right\|)dt\end{aligned}$$

where $t_0 = \min\{\tau_-, \tau_+\}$, $\hat{q}_i(t)$ is the position of R_i at time t using the rule

$$\dot{\hat{q}}_i(t) = \Xi(\hat{q}_i(t), \hat{q}_{i-1}(t), \hat{q}_{i+1}(t)) \quad (29)$$

Moreover,

$$\vec{\tilde{T}}_i(t) = \frac{\hat{q}_{i+1}(t) - \hat{q}_{i-1}(t)}{\|\hat{q}_{i+1}(t) - \hat{q}_{i-1}(t)\|} \quad (30)$$

$$\vec{\tilde{N}}_i(t) = -\begin{bmatrix}1 & 0\end{bmatrix}\vec{\tilde{T}}_i(t)\begin{bmatrix}0\\1\end{bmatrix} + \begin{bmatrix}0 & 1\end{bmatrix}\vec{\tilde{T}}_i(t)\begin{bmatrix}1\\0\end{bmatrix} \quad (31)$$

Qualitatively speaking, η_{N_i} and η_{T_i} try to keep q_i, q_{i-1} and q_{i+1} within certain bounds. Intuitively enough, a reasonable way of bounding $\|q_i(t) - \hat{q}_i(t)\|$ is to bound $\|\hat{q}_i(t) - \hat{q}_{i-1}(t)\|$, $\|\hat{q}_i(t) - \hat{q}_{i+1}(t)\|$ and $\|\hat{q}_{i+1}(t) - \hat{q}_{i-1}(t)\|$ thus preventing a suitably defined energy function to fall below a certain threshold. This denotes an event-triggered synchronisation, as an alternative to traditional time-triggered synchronisation. As long as certain bounds are satisfied, the linear interpolation scheme is a good approximation. In practice, even if the above bounds are satisfied, deviation of $v_{i+1}(t)$ and $v_{i-1}(t)$ from their estimated values at times $t-\tau_+$ and $t-\tau_-$ necessitate synchronization after some time. To decide when this should happen, we can test for the violation of the bounds

$$\begin{aligned}(\hat{v}_{i-1} + \varepsilon)(t + \Delta t - \tau_-) &\le \delta_-\\ (\hat{v}_{i+1} + \varepsilon)(t + \Delta t - \tau_+) &\le \delta_+\end{aligned} \quad (32)$$

where ε is a small positive constant. Figure 4.b shows the hybrid automaton modelling this *semi-asynchronous* scheme.

4. Formation initialization

The robots are initially scattered throughout a region with no definite structure (randomly distributed). Assume that a circular neighborhood $\mathcal{N}_r(q_i(t))$, with radius r, is defined around each robot R_i. This neighborhood can designate the limited communication range of the robots. $R_j \in \mathcal{N}_r(q_i(t))$ if and only if $\|q_i(t) - q_j(t)\| \le r$. A connection graph $\mathcal{G}$ can be constructed in which nodes correspond to robots and an edge e_{ij} is present between two nodes R_i and R_j if $R_j \in \mathcal{N}_r(q_i(t))$ and $R_i \in \mathcal{N}_r(q_j(t))$. We will assume that the initial graph is connected. Before adaptation is attempted, the robots have to self-organize into a formation suitable the problem at hand, i.e., a roughly straight line in the case of open contours and a circle in the case of closed ones. In [6], algorithms are proposed for these two cases. Here, we will adapt their method to the case of real robots with dimensions (instead of points). We also slightly modify and reformulate the original versions. First, note that it is assumed that each robot is equipped with a compass so that it can measure its orientation. Suppose a local coordinate system Υ_i is attached to the robots whose Y_i axis is aligned with direction of the north. All the positions are measured in this local coordinates, with

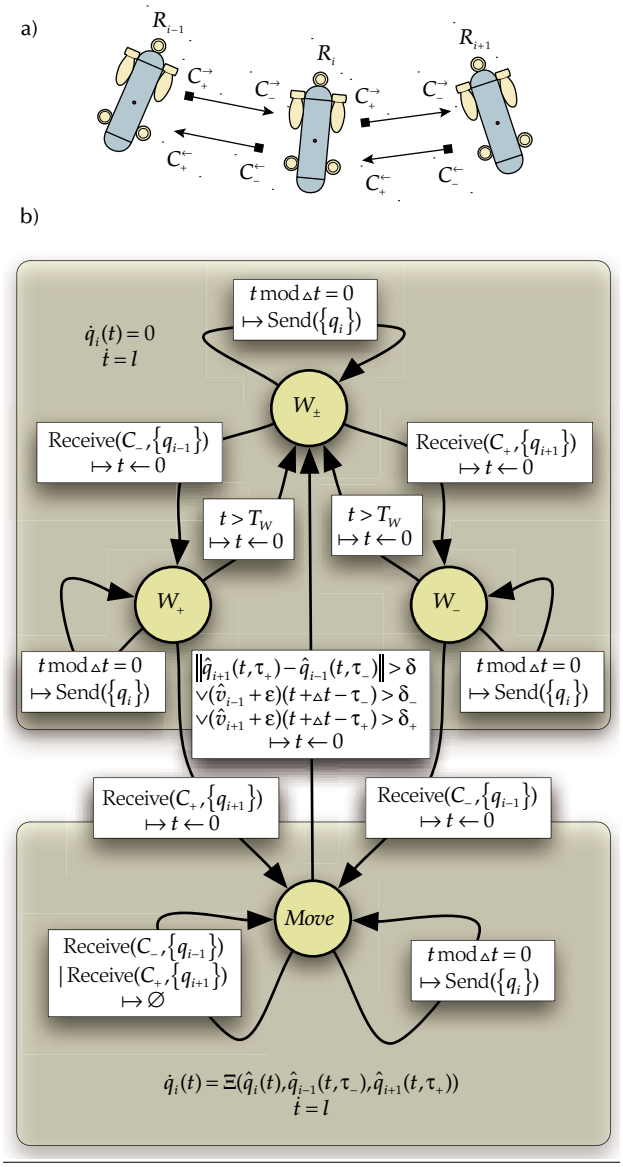

figure 4: a) Communication channels. b) Automaton for semi-synchronous motion.

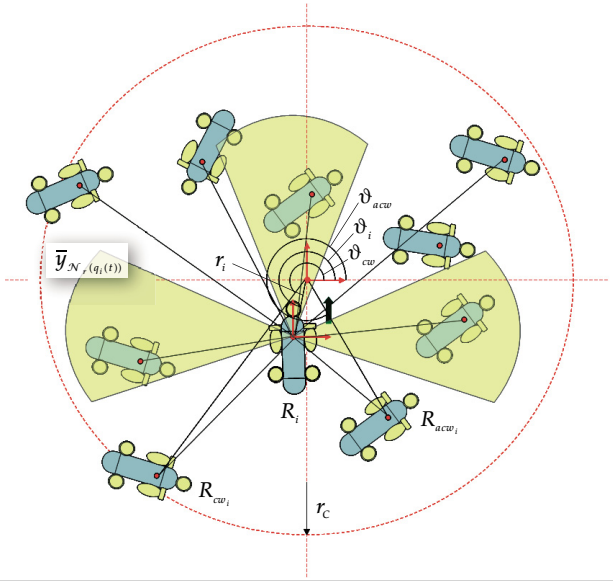

figure 5: Notations for formation initialization

q_i located at the origin. Moreover, assume that each robot has the means to measure the distances and bearings to the robots in its neighborhood. No two robots are allowed to come closer that a distance D_M to each other. Also, let L denote the desired inter-robot distance between a pair of robots on the final line formation. Implementing the proposed algorithm, the final line would spread along the X axis of a global coordinate system Υ whose Y axis is aligned with the north. The position of the X axis along Y direction would be the mean of the y coordinates of all the robots, if measured with respect to this global system.

We define three virtual sensors s_R, s_L, s_{V+} and s_{V-} as follows. s_R indicates if the path of the robot is obstructed in the easterly direction. The other sensors serve the same purpose for west, north, and south directions, respectively. More formally:

Definition 1: *$s_R = 1$ for R_i if there exists a $R_j \in \mathcal{N}_r(q_i(t))$, $R_j \neq R_i$, such that*

$$\|q_i(t) - q_j(t)\| \leq D_M, \tag{33}$$

$$\|q_i(t) - q_j(t)\| = \inf_{R_k \in \mathcal{N}_r(q_i(t)) \wedge \|q_i(t)-q_k(t)\| \leq D_M} \|q_i(t) - q_k(t)\| \tag{34}$$

and

$$|\vartheta_{ij}| = \left| tg^{-1}\left(\frac{y_i - y_j}{x_i - x_j}\right)\right| \leq \vartheta_R \tag{35}$$

where $\vartheta_R = \vartheta_S$. $s_R = 0$ otherwise. ϑ_S is an angle determining the degree of open-ness in a direction. To avoid collisions, ϑ_S should be bigger that $\pi/4$. The other sensors are similarly defined. For s_L, we define $\vartheta_L = \pi - \vartheta_S$. Also, $\vartheta_{V+} = \pi/2 - \vartheta_S$, and $\vartheta_{V-} = 3\pi/2 - \vartheta_S$. We denote R_j by R_i^{χ}, where $\chi \in \{L, R, V, V+, V-\}$.

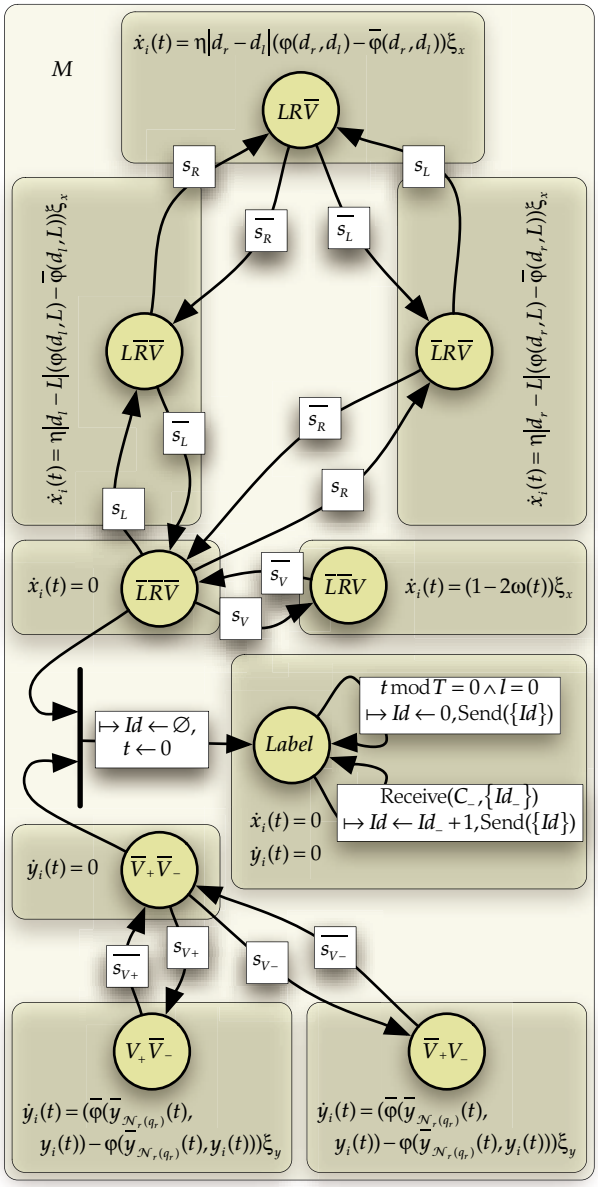

figure 6: Automaton for formation initialization. M is composed of two automatons, run in parallel, for motions in x and y directions. States are named according to the states of sensors. Also, $\xi_x = \gamma_x(\|\bar{y}_{\mathcal{N}_r q_i} - y_i\|)v_x$ and $\xi_y = v_y/\gamma_y(\|\bar{y}_{\mathcal{N}_r(q_i)} - y_i\|)$.

Now, let each robot move according to the following control rule:

$$\dot{q}_i(t) = [\dot{x}_i(t), \dot{y}_i(t)]^T \tag{36}$$

$$\begin{aligned}\dot{x}_i(t) = {} & (s_L s_R \eta(|d_r - d_l|)(\phi(d_l, d_r) - \bar{\phi}(d_l, d_r)) \\ & - (s_L \bar{s}_R \eta(|d_l - L|)(\phi(d_l, L) - \bar{\phi}(d_l, L)) \\ & + \bar{s}_L s_R \eta(|d_r - L|)(\bar{\phi}(d_r, L) - \phi(d_r, L)) \\ & + \bar{s}_L \bar{s}_R s_V (1 - 2\omega(t)))\gamma_x(\|\bar{y}_{\mathcal{N}_r(q_i)} - y_i\|)v_x)\end{aligned} \tag{37}$$

$$\begin{aligned}\dot{y}_i(t) = {} & \bar{s}_V(\bar{\phi}(\bar{y}_{\mathcal{N}_r(q_i)}(t), y_i(t)) \\ & - \phi(\bar{y}_{\mathcal{N}_r(q_i)}(t), y_i(t)))(1/(\gamma_y(\|\bar{y}_{\mathcal{N}_r(q_i)} - y_i\|)))v_y\end{aligned} \tag{38}$$

where

$$\bar{y}_{\mathcal{N}_r(q_i)}(t) = \frac{1}{n(\mathcal{N}_r(q_i))}\sum_{j=1}^{n(\mathcal{N}_r(q_i))} y_i(t) \quad (39)$$

is the Y coordinate of the centre of gravity of robots in $\mathcal{N}_r(q_i)$,

$$s_V = s_{V+}\bar{\phi}(\bar{y}_{\mathcal{N}_r(q_i)}(t), y_i(t)) + s_{V-}\phi(\bar{y}_{\mathcal{N}_r(q_i)}(t), y_i(t)) \quad (40)$$

$\bar{u}$ denotes the boolean negation of the boolean variable u, $\omega(t)$ is a binary random variable, $d_r = \|q_i(t) - q_i^R(t)\|$, $d_l = \|q_i(t) - q_i^L(t)\|$, $q_i^L(t)$ and $q_i^R(t)$ being the positions of the robots on the left and right of R_i. $\phi(u_1, u_2)$ is 1 if $u_1 < u_2$ and 0 otherwise. v_x and v_y are nominal constant speeds in the respective (east-west, south-north) directions. $\eta(u)$ monotonically goes to zero as $u \to 0$. $\gamma_x(u)$ is a monotonic function which increases as $u \to 0$. Likewise, $\gamma_y(u)$ is a monotonically decreasing function as $u \to 0$. The intuition behind the use of γ_x and γ_y is that as the robot approaches the line, its horizontal velocity should increase while its vertical speed should decrease. This cooperative behaviour will help the faster convergence of the robots to the line.

At each instant of time, the error of the formation is defined in [6] as

$$q_e^L(t) = \sum_{i=0}^{N-1} q_{e_i}^L(t) = \begin{pmatrix} x_{e_i}(t) \\ y_{e_i}(t) \end{pmatrix} = \bar{q}(t) + \begin{pmatrix} \frac{N+2i-1}{2}L \\ 0 \end{pmatrix} - q_i(t) \quad (41)$$

Note that the line formation strategy can also be used to form lines rotated with respect to Υ. For initialization of closed contour formations, the robots should form a circle with an appropriate radius (say r_C) around the centroid of the initial assembly. In this paper, as we are focusing on open contours, we will not elaborate on realistic control laws for this case. Assuming that robots are point particles and collisions are not an issue, the following control rule will achieve the circular formation:

$$\dot{q}_i(t) = \begin{pmatrix} \dot{x}_i(t) \\ \dot{y}_i(t) \end{pmatrix} = \begin{pmatrix} \cos(\vartheta_i(t)) & -\sin(\vartheta_i(t)) \\ \sin(\vartheta_i(t)) & \cos(\vartheta_i(t)) \end{pmatrix} \times \begin{pmatrix} \phi(r_i(t), r_C) - \bar{\phi}(r_i(t), r_C) \\ \bar{\phi}(\vartheta_{acw_i}(t), \vartheta_{cw_i}(t)) - \phi(\vartheta_{cw_i}(t), \vartheta_{cw_i}(t)) \end{pmatrix}^T \begin{pmatrix} v_R \\ v_T \end{pmatrix} \quad (42)$$

where v_R and v_T denote, respectively, nominal *radial* and *tangential* speeds,

$$\bar{q}(t_0) = \frac{1}{N}\sum_{i=0}^{N-1} q_i(t_0) \quad (43)$$

(t_0 being the initial time),

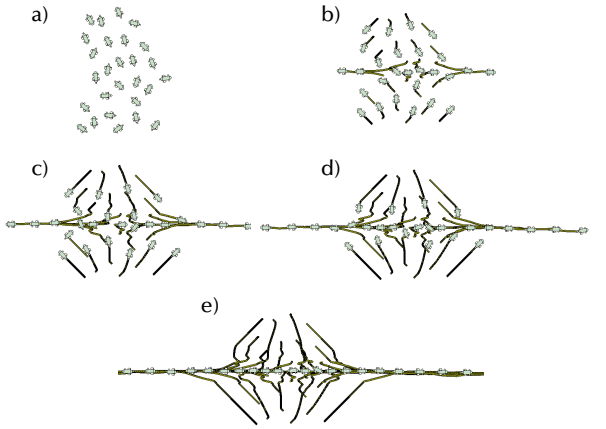

figure 7: Formation initialization

$$r_i(t) = \|\bar{q}(t_0) - q_i(t)\|,$$
$$\vartheta_i(t) = tg^{-1}\left(\frac{y_i(t) - y(t_0)}{x_i(t) - x(t_0)}\right). \quad (44)$$

acw_i and cw_i denote the indexes of counter-clockwise and clock-wise immediate neighbors of R_i, respectively. The formation error can now be defined by

$$q_e^C(t) = \sum_{i=0}^{N-1} q_{e_i}^C(t) \quad (45)$$

$$q_{e_i}^C(t) = \begin{pmatrix} r_{e_i}(t) \\ \vartheta_{e_i}(t) \end{pmatrix} = \begin{pmatrix} \bar{r}(t) \\ \bar{\vartheta}(t) \end{pmatrix} + \begin{pmatrix} r_C \\ -\frac{1}{N}\pi(N-1) + \frac{1}{N}2\pi i \end{pmatrix} - \begin{pmatrix} r_i(t) \\ \vartheta_i(t) \end{pmatrix} \quad (46)$$

Finally, an efficiency measure is introduced in [6] as

$$\text{eff}(t) = \frac{1}{n}\sum_{i=0}^{N-1} \frac{\|q_i(t) - q_i(t_0)\|}{l_i(t)} \quad (47)$$

where

$$l_i(t) = \int_{t_0}^{t} \dot{q}_i(\tau)\tau \quad (48)$$

is the total distance travelled by R_i. After the formation of the line, robots have to re-label themselves to define a proper parametrization for the contour. Figure 6 shows the hybrid automaton for the line formation process and subsequent re-labelling.

5. Simulation results

First, we present a simulation run for formation initialization. It should be noted that the success of this process depends on the parameters involved: the radius of the neighborhood r, the desired final distance between robots in the final formation L, the minimum allowable distance between two robots D,

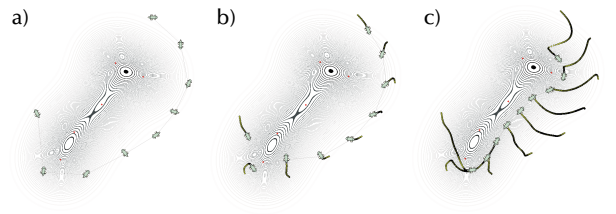

figure 8: Iso-cline adaptation

the nominal speeds v_x and v_y, the rates of change of γ_x and γ_y, and the random variable ω. r should be large enough to ensure connectivity and convergence. In our experiments, it was found that L should be sufficiently larger that D. Setting $L = 2D$ gives good results. γ_x continuously goes from $1/4$ to 1, as $\left\|\bar{y}_{\mathcal{N}_r(q_i)} - y_i\right\|$ goes from r to 0. Likewise, γ_y continuously goes from 1 to $1/4$, as $\left\|\bar{y}_{\mathcal{N}_r(q_i)} - y_i\right\|$ goes from r to 0. To prevent the robot motion from jittering when close to the line, the rate of change of $\omega(t)$ should be small. To make this happen, it is enough to retain the last value of the random variable for some time, i.e., the robot will move according to the value of $\omega(t)$, determined at time instant t, for the duration of $[t, t + \delta t]$. Figure 7 shows the sequence of motions of an initial random aggregate.

Figure 8 shows the sequence of snapshots of the adaptation of a group of robots to a level set of an imaginary field. The red circles show the sites from which the plume has been released.

Figure 9.a shows the adaptation of a group of robots which had previously initialized into a line oriented with respect to the major axis of the field. Figure 9.b shows the adaptation when the initial line overlaps with the field. This latter case is the most problematic. The final shape depends on the shape of the plume and is very difficult to determine beforehand.

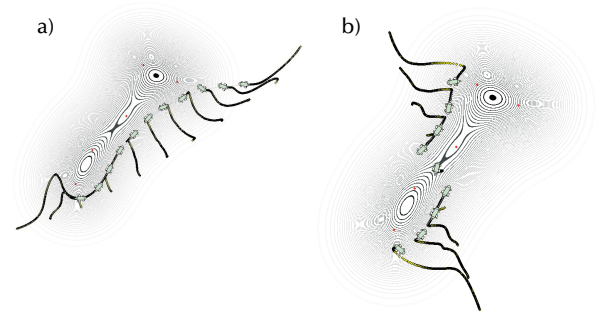

figure 9: Iso-cline adaptation. (a) Adapting from a line oriented with respect to the major axis of the field. Note that it takes much longer for the right most part to adapt. (b) Adapting from a line initially overlapping the field. As can be seen from the traces, inter-robot distances are preserved at all times.

6. Conclusions and future research

We showed that a formation shaped as a contour can adapt itself to areas of equal concentration produced when some chemical is diffused into an environment. The proposed scheme is *scalable* and there is no need for a global reference frame. The measurements and decision making are with respect to local coordinate systems fixed on each robot. These coordinates are aligned with the tangent to the curve and the normal to it. We also discussed synchronous and semi-synchronous implementation methods, as well as algorithms for the initialization of a random aggregate into suitable shapes. There are still some issues which have to be addressed. We assumed that each robot can measure the local gradient at its position. A more realistic approach would be to design contour formations of rigid formations, instead of single vehicles, which can collectively estimate the gradient. It was assumed that the environment is obstacle-free which is rarely the case. Finally, the motion of the formation is confined to a plane parallel to the ocean bottom. Studying motion constrained to surfaces as well as in three dimensions are among possible extensions to the work presented here.

References

[1] Kalantar, S., Zimmer, U., *Contour Shaped Robotic Formations for Isocline Adaptation*, Proc. of the intl. conf TAROS '05, London, U.K., 2005.

[2] Okubo, A., *Diffusion and Ecological Problems: Mathematical Models*, Vol. 10 in Biomathematics Series, Springer-Verlag, 1980.

[3] M. Egerstedt, X. Hu: *Formation Constrained Multi-Agent Control*, IEEE Trans. on Robotics and Automation, Vol. 17, No. 6, Dec. 2001, pp. 947-951.

[4] Caselles, V., Kimmel, R., Sapiro, G., *Geodesic Active Contours*, International Journal of Computer Vision 22(1), 1997.

[5] Chaimowicz, L., Michael, N., Kumar, V., *Controlling swarms of robots using interpolated implicit functions*, Proc. IEEE Conf. ICRA 2005.

[6] Lee, J., Venkatesh, S., Kumar, M., *Formation of a geometric pattern with a mobile wireless sensor network*, Journal of Robotic Systems, Volume 21, Issue 10, October 2004.

[7] Ogren, P., Fiorelli, E., Leonard, N.E., *Formations with a Mission: Stable Coordination of Vehicle Group Maneuvers*, Proc. Symp. on Mathematical Theory of Networks and Systems, 2002.

[8] R.Olfati-Saber and R.M.Murry: *Distributed Cooperative Control of Multiple Vehicle Formations Using Structural Potential Functions*, The 15th IFAC World Conf., June 2002.

[9] Marthaler, D., Bertozzi, A. L., *Tracking environmental level sets with autonomous vehicles*, Recent Developments in Cooperative Control and Optimization, Kluwer, 2004.

[10] Fossen, T.I., *Guidance and Control of Ocean Vehicles*, John Wiley and Sons, 1994.