Particle swarm control

Pierre Allain, Nicolas Courty, Thomas Corpetti

Particle Swarm Control

P. Allain, N. Courty, T. Corpetti

Université de Bretagne Sud, rue Yves Mainguy, 56000 Vannes, France

SUMMARY

Controlling several and possibly independent moving agents in order to reach global goals is a tedious task that has applications in many engineering fields such as robotics or computer animation. Together, the different agents form a whole called swarm, which may display interesting collective behaviors. When the agents are driven by their own dynamics, controlling this swarm is known as the particle swarm control problem. In that context, several strategies, based on the control of individuals using simple rules, exist. This paper defends a new and original method based on a centralized approach. More precisely, we propose a framework to control several particles with constraints either expressed on a per-particle basis, or expressed as a function of their environment. We refer to these two categories as respectively *Lagrangian* or *Eulerian* constraints. The contributions of the paper are the following: *i)* we show how to use optimal control recipes to express an optimization process over a large state space including the dynamic information of the particles; *ii)* the relation between the Lagrangian state space and Eulerian values is conveniently expressed with graph operators that make it possible to conduct all the mathematical operations required by the control process. We show the effectiveness of our approach on classical and more original particle swarm control problems.

KEY WORDS: Particle Swarm Control ; Optimal control ; Variational assimilation ; Adjoint method ; Graph operators

1. INTRODUCTION AND MOTIVATIONS

This paper focuses on the control of a set of particles which together form a whole, the latter being called "swarm". From individual properties, and depending on the nature and the state of particles, collective behavior may appear from such entity [1, 2]. Controlling these collective behaviors together with individual particles is therefore a key issue which is explored in this article. This problem can be found in the literature for several distinct application fields, such as robot swarms [3, 4], simulation of schools and herds of animals [5], human crowd simulation [6] or even mobile networks with switching topology [7]. The control of a particle swarm has mainly been focused on three different problems: *i)* the *rendezvous* problem, also known as the distributed consensus problem [7, 8], which aims at agreeing on and reaching a common location, *ii)* the *formation* problem whose goal is to keep a given formation between the different swarm members [9, 10] and finally the *iii) aggregation* problem, where the objective is for the agents to group in a cohesive swarm [11, 12]. Other issues arise from economic load dispatching [13] which looks for the optimal configuration of generators. Most of the previous works on particle swarm control focus on the design of individual laws, usually based on potential fields (see e.g. [14, 15]), to define the control policy. The latter induces an emerging and global behavior of the swarm also known as *the swarm intelligence*, which has raised several important stability issues [11, 16, 17]. A good review of those problems and the common associated recipes, along with stability results, can be found in [15]. In fact, those decentralized approaches performed at the particle level are motivated by real world conditions (i.e. physical systems which can only communicate with their local neighbors), but also because the global control problem is tedious, since it requires to minimize non-convex energy functions, as in [18], defined over potentially very large state space, with possibly chaotic behaviors. The particle swarm optimization (PSO) methods [19], of which a good overview can be found in [20], allows the minimization of such functionals using sampling

methods for gradient descent, but is different in nature of the problem addressed in this paper. We notably propose a new methodology where the swarm control operates globally to enforce a desired consistency, and for which the gradient of the functional is computed, unlike the PSO methods. In order to fit to the different problems' nature above cited, the control objectives can be expressed either on a per-particle basis or relatively to the environment. These two categories of aims are illustrated in Fig. 1 and will be respectively referred to as *Lagrangian* and *Eulerian* constraints or goals. However, finding an explicit correspondence between Lagrangian data and Eulerian quantities (e.g. position of particles and its analog quantity density) is a non trivial task and calls for a dedicated methodology.

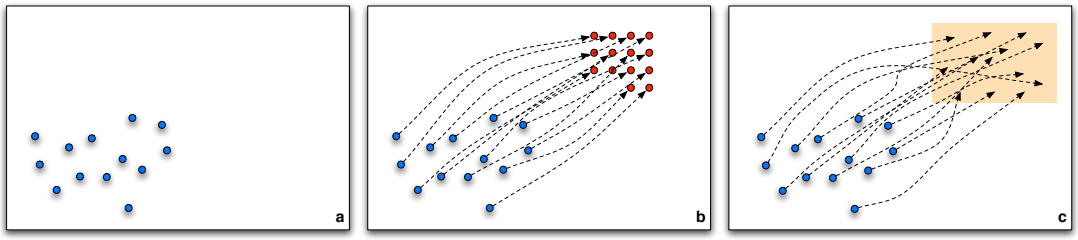


Figure 1. Illustrations of the different nature of constraints in the swarm control problem. From an initial configuration (a), particles in blue can be controlled either to reach individual positions represented as red circles: *Lagrangian* constraints (b) or to meet on a given area: *Eulerian* constraint

Assuming that the particles are driven by an evolution model expressed as a system of partial differential equations (PDE) (a generic model is given in Section 3 to illustrate our control method), we propose to use a deterministic optimization technique from the optimal control family [21, 22]: the variational assimilation framework. Mostly based on the adjoint theory, it has been widely used the last three decades [23, 24], mainly because of its convenient way to express the gradient of a cost-function involving a large system state and the associated parameters. This method has most of the time been used in a pure Eulerian framework [25, 26] but some applications in the context of Lagrangian setpoints can be found [27].

The contributions of the paper are then twofold:

- we provide an original theoretical framework which makes it possible to derive a correspondence between the Lagrangian and Eulerian space (Section 4) of quantities related to the swarm. This mapping is performed generically thanks to operators inspired from the graph theory [28]. Those operators allow us to conduct mathematical operations such as the linearization of the constraint operators required by the variational assimilation process (Section 5),
- we recast the centralized swarm control problem within the optimal control theory, and show the potential benefits of this approach by solving several classical particle swarm control problems (Section 6). Among these, we provide formation or aggregation control, and less classical issues such as control with respect to high order dynamical quantities such as divergence and vorticity of the swarm velocity.

Finally, a discussion and a conclusion end the paper (Section 7).

2. NOTATIONS AND METHOD OVERVIEW

We first begin by introducing in this section the terminology and the notations used in the remainder of the paper. The control of a swarm of particles requires three main key points:

- **the way particles behave in a deterministic fashion**: *i.e.* how they move together subject to given parameters. More formally, this corresponds to the model $\mathbb{M}$, which drives the state of all particles $\mathbf{X}$ through the following evolution equation:

$$\frac{\partial \mathbf{X}}{\partial t} + \mathbb{M}(\mathbf{X}) = 0; \tag{1}$$

- **external constraints applied to the swarm:** the setpoints $\mathbf{Y}$. These constraints can be expressed in a different space than $\mathbf{X}$ as it is the case in the swarm control problem considered in Fig. 1.c, where the particle density is used as a continuous constraint. In that context, ad-hoc relations capable of linking the Lagrangian system space to such constraints have to be defined. As explained later in the paper, this is performed through graph operators which formalism is very flexible and valid for dimensions $\dim \in \{1, 2, 3\}$;
- **the swarm control**, thanks to the mathematical process combining the last two concepts. In this paper, we apply the control theory to a system deriving from $\mathbb{M}$ to obtain specific states $\mathbf{X}(t)$. A sensor is used to compare the system output to the reference signal. This error is operated by a controller giving appropriate control variables for the system. Its integration will lead to a corrected output. In our case, the controller corresponds to the **assimilation process**, the system to the evolution model $\mathbb{M}$, and the sensor to the observation operator $\mathbb{H}$. We note $\mathbf{Y}$ the reference signal, which can be considered as an observation in an *estimation* framework, or as a setpoint in the control approach presented in this paper. Fig. 2 shows the negative feedback control of the state with a user defined signal. This control loop is

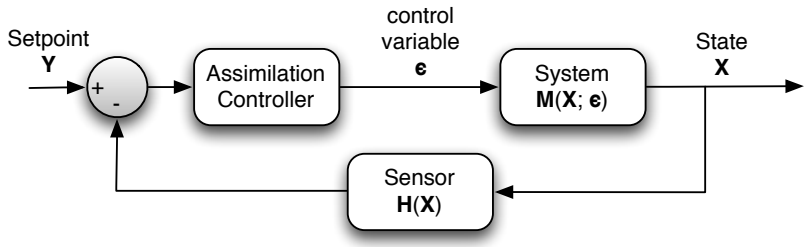


Figure 2. **Functional scheme of the method**. Each iterative loop computes the difference between the setpoint $\mathbf{Y}$ and the observed state $\mathbb{H}(\mathbf{X})$ in the setpoint space. The difference supplies the assimilation process deriving from the model $\mathbb{M}$ which will give new states $\mathbf{X}(t)$ all along time.

not expressed in the frequency domain, and $\mathbf{X}$ cannot be determined by one-pass analytical means unlike many engineering control applications. The reason is the specificity of both the assimilation controller and the system which contain time integration of equations.

3. SWARM EVOLUTION MODEL

To demonstrate the ability of the method to stick with common physical systems, we chose to consider a second order model, abiding by Newton's law of motion. This choice is motivated by the large variety of swarming phenomena these equations are able to model, like autonomous mobile robots [29, 30], crowds dynamics [6], or schools of fish [31]. We obtain for each particle p_i the system:

$$\begin{cases} \dfrac{\partial y_i}{\partial t} & = u_i, \quad (2a) \\ m_i \dfrac{\partial u_i}{\partial t} & = \mathbb{F}_i. \quad (2b) \end{cases}$$

In this model, an overall force $\mathbb{F}_i$ term is applied to every particle p_i. We propose to decompose this force into three main components:

$$\mathbb{F}_i = \mathbb{F}_{\text{source},i} + \mathbb{F}_{\text{slow},i} + \mathbb{F}_{\text{interaction},i} \quad (3)$$

The source term provides a uniform and constant displacement, and is composed by a direction W_i and an intensity α_i given for every particle. This term reads:

$$\mathbb{F}_{\text{source},i} = \alpha_i W_i \tag{4}$$

In order to limit the total kinetic energy of the system, it is common to add a friction force reading for every particle p_i:

$$\mathbb{F}_{\text{slow},i} = -k_i u_i. \tag{5}$$

Providing an interaction term in the model demonstrates the power of the assimilation process in reverberating corrections among a complex system like interactive swarm. One of the most simple and yet potentially useful interactions is the repulsive force which has been widely used to model social respect for both humans and animals. We consider it as decreasing with the distance between particles and so take it as an inverse exponential function [6], and directed toward the particle p_i:

$$\boldsymbol{r}_{ij}^{\mathbf{y}} = -ae^{-b\|y_j - y_i\|}e_{ij}. \tag{6}$$

It is clear that the above model dynamics is described at a particle level, which renders a global description at a swarm level difficult to achieve. It is also indeed difficult to consider at this stage high level constraints involving global properties (such as density or velocity). The purpose of the next section is to present a new formalism based on graph operators aiming at bridging the gap between those different levels of representations.

4. GRAPH REPRESENTATION

Formalizing Lagrangian dynamics requires to define equations at a particle level. When one manipulates physical systems associated with continuous laws, such definition can be tedious. Here we propose to re-write such dynamics directly at the swarm level. The benefit of such approach is to define inner swarm interactions. Furthermore, the graph formalism along with its accompanying matrix provides excellent tools for derivation of Lagrangian functions. The definitions of such operators and Jacobian matrix needed by the variational assimilation, are suggested in this section. We also propose to explicitly link particle data to discretized continuous quantities by viewing an Eulerian grid as another graph. Neighboring conditions for particles are also defined thanks to these operators. Before entering into details, let us now present some generalities.

4.1. Generalities

We consider the swarm as a set of N particles, seen as vertices. We note $\mathbb{V}$ the vertex space and $\mathbb{E}$ the edge space.

The swarm $\mathcal{V} \in \mathbb{V}^N$ has interconnections seen as edges $\mathcal{E} \in \mathbb{E}^Z$, where Z is the number of edges connecting two vertices of $\mathcal{V}$. In practice obviously we are not interested in connecting every particle to all others but we are rather interested on some local interaction. Therefore we have $Z \ll N^2$. We obtain the definition of the graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ defining in a row particles and their interactions. It is important to note that each vertex carries different properties or quantities, usually denoted as: $\begin{array}{rccc} \mathbf{f}: & \mathbb{V}^N & \rightarrow & \mathbb{F}^N \\ & \mathcal{V} & \mapsto & \mathbf{f}(\mathcal{V}) \end{array}$, with $\mathbb{F} = \mathbb{R}$ in the case of a scalar quantity. For example, the position of particles, also called configuration, can be written as:

$$\mathbf{y}(\mathcal{V}) = \begin{bmatrix} y(p_1) \\ \dots \\ y(p_i) \\ \dots \\ y(p_N) \end{bmatrix} = \begin{bmatrix} y_1 \\ \dots \\ y_i \\ \dots \\ y_N \end{bmatrix}. \tag{7}$$

In order to write graph quantity expressions, we introduce the diagonalisation operator $\underline{.}$ which creates a pure diagonal matrix made of the vector values. For instance, $\underline{\mathbf{f}}\mathbf{y}$ gives us a new graph quantity being the product of $\mathbf{f}$ and $\mathbf{y}$.

4.2. Connectivity

The connectivity of vertices $\mathcal{V}$ is defined by the adjacency matrix $\boldsymbol{A} \in \mathbb{Z}^{N\times N}$, where $\mathbb{Z}$ is the integer domain, which captures the structure of the graph $\mathcal{G}$. This matrix is defined as:

$$\boldsymbol{A} = \begin{cases} \boldsymbol{A}_{ij} = 1 & \text{if there exists an edge between } p_i \text{ and } p_j, \\ \boldsymbol{A}_{ij} = 0 & \text{if there is no edge.} \end{cases} \tag{8}$$

In most cases, relations between particles decrease with the distance: the connectivity depends on the proximity. There exist several ways to set the proximity-based connectivity, such as connecting vertices within a range, or connecting the k nearest neighbors. Because of the potentially high number of particles, we chose to use a neighboring condition in a regular grid embedding, which makes it possible to perform computations more efficiently only for a subset of neighboring particles. The adjacency matrix often goes with the degree matrix $\boldsymbol{D} \in \mathbb{Z}^{N\times N}$, which accounts for the sum of connections for every vertex. This matrix is diagonal and is defined as:

$$\boldsymbol{D} = \begin{cases} \boldsymbol{D}_{ii} = \sum_j \boldsymbol{A}_{ij}, \\ \boldsymbol{D}_{ij} = 0. \end{cases} \tag{9}$$

Let us now take the example in a two dimension space of a given configuration of vertices with a first order neighboring condition described in Fig. 3.

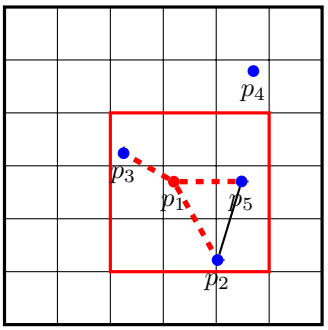


Figure 3. **Example graph** $\mathcal{G}$. The swarm lies on an Eulerian grid $\mathcal{C}$. Connectivity of particles is here defined by meshes proximity with emphasis on node p_1.

The adjacency and degree matrix of graph $\mathcal{G}$ are:

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{D} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}. \tag{10}$$

In order to express explicitly the adjacency based on a Eulerian grid, we introduce the mesh belonging matrix $\mathcal{P}(\mathcal{V}) \in \mathbb{Z}^{N\times M}$, where M is the number of meshes c of grid $\mathcal{C}$. $\mathcal{P}$ states for the presence of a vertex in a mesh. We define:

$$\mathcal{P} = \begin{cases} \mathcal{P}_{im} = 1 & \text{if } y_i \text{ is inside mesh } c_m, \\ \mathcal{P}_{im} = 0 & \text{otherwise.} \end{cases} \tag{11}$$

Simultaneously we introduce the grid adjacency matrix $\mathcal{M}_o \in \mathbb{Z}^{M\times M}$ representing the connectivity of meshes at o-th order. We define:

$$\mathcal{M}_{o,ml} = \begin{cases} 1 & \text{if } \|c_m - c_l\|_\infty \leq o, \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

with $c_m - c_l$ the leap vector to go from mesh c_l to c_m. The combination of these two matrices gives us the precise expression of the adjacency:

$$\boldsymbol{A} = \mathcal{P}\mathcal{M}_o\mathcal{P}^T. \tag{13}$$

As for the example in Fig. 3, one can easily see that $\mathcal{P}^T$ gives the mesh to which p_1 belongs, $\mathcal{M}_1$ gives the first order neighbours of this mesh, then $\mathcal{P}$ collects vertices belonging to this set of meshes.

4.3. Weighted relations

Even if the conditions of the connectivity are known, the intensity of the connection must be known as well, since functionals will have to depend on and eventually drive vertices. Thus, the adjacency matrix can be augmented by weights. These weights $w(f(p_i), f(p_j)) \in \mathbb{W}$, or $w^{\mathbf{f}}_{ij}$, create the matrix $\left[w^{\mathbf{f}}_{ij}\right] \in \mathbb{W}^{N\times N}$ which when combined to the adjacency matrix leads to:

$$\boldsymbol{A}_{w^{\mathbf{f}}} = \left[w^{\mathbf{f}}_{ij}\right] \circ \boldsymbol{A}, \tag{14}$$

$\circ$ denoting the Hadamard, or entrywise, product. In practice, the quantity used by the weight w_{ij} is often a difference of the positions y_i and y_j, reading: $w^{\mathbf{y}}_{ij} = w(y(p_j) - y(p_i))$. As an example, if taking the weight function as a Gaussian kernel $\boldsymbol{G}$ parameterized by its standard deviation σ, such as:

$$\boldsymbol{G}^{\mathbf{y}}_{ij} = \boldsymbol{G}(y_i, y_j) = \frac{1}{\left(\sigma\sqrt{2\pi}\right)^{\dim}} e^{-\frac{(y_j-y_i)^2}{2\sigma^2}}, \tag{15}$$

one can express in a compact way the concentration of the particle's quantity, or particle density, of $\mathcal{G}$ by writing:

$$\boldsymbol{\rho} = \boldsymbol{A}_{\boldsymbol{G}^{\mathbf{y}}}\mathbf{q}, \tag{16}$$

with the quantity of particle $\mathbf{q}(\mathcal{V})$ here set to $\vec{1}$, meaning every particle counts for 1 ptl*. Consequently the dimension of the concentration becomes: $[\boldsymbol{\rho}] = ptl.m^{-\dim}$. An example of this density is proposed in Fig. 5 *(left)*.

4.4. Rewriting the evolution model with graph operators

At this stage, it is possible to rewrite the evolution model exposed in Section 3 with graph operators. Recalling that every particles are submitted to a force $\mathbb{F}_i$ composed of three terms $\mathbb{F}_i = \mathbb{F}_{\text{source},i} + \mathbb{F}_{\text{slow},i} + \mathbb{F}_{\text{interaction},i}$, we rewrite the three different components in the following way:

- *the source term*, which is is composed by a direction $\mathbf{W}(\mathcal{V}) \in \mathbb{R}^{d\times N}$ and an intensity $\boldsymbol{\alpha}(\mathcal{V}) \in \mathbb{R}^N$ now reads in the graph's space: $\mathbb{F}_{\text{source}} = \underline{\boldsymbol{\alpha}}\mathbf{W}$;
- *the friction term*, which is composed by the particles velocities $\mathbf{u} \in \mathbb{R}^{d\times N}$ and the friction coefficients $\mathbf{k} \in \mathbb{R}^N$ nows reads in the graph's space: $\mathbb{F}_{\text{slow}} = -\underline{\mathbf{k}}\mathbf{u}$;
- *the interaction term*, which is composed of weighted relations between particles now reads in the graph's space: $\mathbb{F}_{\text{interaction}} = \boldsymbol{A}_{\boldsymbol{r}^{\mathbf{y}}}\vec{1}$;

which finally gives the following expression of the evolution model in the graph's space:

$$\begin{cases} \dfrac{\partial \mathbf{y}}{\partial t} & -\mathbf{u} = 0, \qquad (17a) \\ \dfrac{\partial \mathbf{u}}{\partial t} & -\underline{\mathbf{m}}^{-1}(\underbrace{\underline{\boldsymbol{\alpha}}\mathbf{W} - \underline{\mathbf{k}}\mathbf{u} + \boldsymbol{A}_{\boldsymbol{r}^{\mathbf{y}}}\vec{1}}_{\mathbb{F}}) = 0, \qquad (17b) \end{cases}$$

in which one can recognize an evolution equation as defined at Eq. 1. We now examine the possibilities offered by the graph operators to perform a link between the Lagrangian quantities

*We note ptl the unit of the quantity of particles.

involved in the graph and Eulerian quantities relative to the environment. We recall again that this can be useful if one wants to control the swarm with respect to constraints given at the swarm level such as density or velocity.

4.5. Projection to Eulerian space

In addition to the grid neighboring condition, we also need to evaluate Lagrangian data in the Eulerian space for the integration of continuous constraints in the control process. Knowing any vertex quantity at the vertex location, we have to presume the values elsewhere, such as on the grid $\mathcal{C}$. We then need to estimate either the concentration of the quantity, or directly the quantity which will act as an interpolation. Of course, every vertices will not be used for the estimation, and we will use $\mathcal{M}_o\mathcal{P}^T$ to select vertices on sight. The weights, however, are slightly different from those used for graph adjacency. They mix the quantity of the vertices and the grid. By defining $\left[w^{\mathbf{f}}_{mj}\right]_{\mathcal{CG}} \in \mathbb{W}^{M\times N}$ the matrix of weights, with $w^{\mathbf{f}}_{mj} = w(f(p_j) - f(c_m))$, we can express the convolution, or concentration here, of $\mathbf{q}$, the density of particles on $\mathcal{C}$ as follows:

$$\boldsymbol{\rho}_{\mathcal{C}} = \left[\boldsymbol{G}^{\mathbf{y}}_{mj}\right]_{\mathcal{CG}} \circ \mathcal{M}_o\mathcal{P}^T\mathbf{q}, \tag{18}$$

provided $y(c_m) = \mathbf{x}_{c_m}$, the center of mesh c_m. In the case of concentration, the weight has to be a kernel function, such as gaussian $\boldsymbol{G}$. The weight reads in this case:

$$\boldsymbol{G}^{\mathbf{y}}_{mj} = \boldsymbol{G}(\mathbf{x}_m, y_j) = \frac{1}{\left(\sigma\sqrt{2\pi}\right)^{\dim}} e^{-\frac{(y_j - \mathbf{x}_m)^2}{2\sigma^2}}. \tag{19}$$

Introducing the Eulerian adjacency matrix $\mathcal{A}_{w^{\mathbf{f}}} \in \mathbb{W}^{M\times N}$ such as:

$$\mathcal{A}_{w^{\mathbf{f}}} = \left[\boldsymbol{G}^{\mathbf{y}}_{mj}\right]_{\mathcal{CG}} \circ \mathcal{M}_o\mathcal{P}^T, \tag{20}$$

Eq. 18 can be simplified as follows:

$$\boldsymbol{\rho}_{\mathcal{C}} = \mathcal{A}_{\boldsymbol{G}^{\mathbf{y}}}\mathbf{q}. \tag{21}$$

We present in Fig. 4 an example of calculation of the concentration of $\mathbf{q}$ in a one dimension space, and in Fig. 5 *(right)* the calculation in a two dimension space.

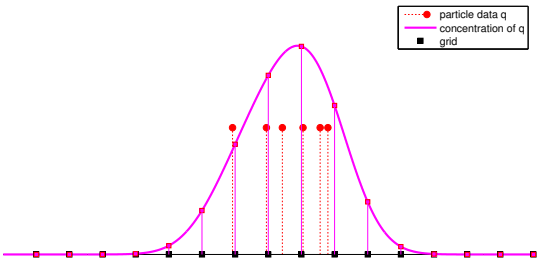


Figure 4. **Concentration example**. Concentration of $\mathbf{q}$ on grid $\mathcal{C}$ obtained by $\mathcal{A}_{\boldsymbol{G}^{\mathbf{y}}}\mathbf{q}$. Here $N = 6$, $M = 16$, $o = M$, and the standard deviation of $\boldsymbol{G}$ is set as the size of the meshes of $\mathcal{C}$.

To get the quantity value elsewhere other than on vertices, we use the Eulerian adjacency weights as interpolation weights. For each p_i, we divide the convolution by the sum of weights applied. This normalization can be done with the introduction of the Eulerian degree matrix $\mathcal{D} \in \mathbb{W}^{M\times M}$, defined as:

$$\mathcal{D} = \begin{cases} \mathcal{D}_{mm} = \sum_j \mathcal{A}_{mj}, \\ \mathcal{D}_{ml} = 0. \end{cases} \tag{22}$$

In the end, letting $\mathbf{u}_{\mathcal{C}} \in \mathbb{R}^{\dim M}$ be the Eulerian velocity expressed on grid $\mathcal{C}$, we obtain the expression:

$$\mathbf{u}_{\mathcal{C}} = \mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{A}_{w^{\mathbf{y}}}\mathbf{u}. \tag{23}$$

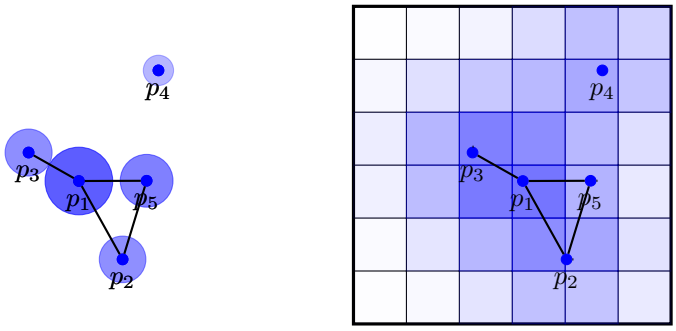


Figure 5. **Example graph** $\mathcal{G}$. *Left)* The density is expressed on graph $\mathcal{G}$ by Eq. 16, *right)* the density is expressed on grid $\mathcal{C}$ by Eq. 21.

This acts as an inverse distance weighted interpolation where the number of samples used equals $\mathcal{M}_o\mathcal{P}^T\vec{1}$, and provided $w^{\mathbf{y}}_{mj}$ decreases as $\|y_j - \mathbf{x}_{c_m}\|$ grows.

The choice of w is critical, and can lead to very different results. It has to be chosen and configured with respect to the expected smoothing degree, and the desired extrapolation type when away from the swarm. Ideally, the perfect w should match the following requirements:

- accuracy: the evaluated quantity at an available data is itself. This reads: $(y_j = \mathbf{x}_{c_m} \Rightarrow u_j = u_{c_m}) \Leftrightarrow w(0) = \infty$;
- vacuum relevance and stability: when far from the available data (i.e. the swarm), the evaluated quantity corresponds to the average of the available data;
- smoothness: as we deal with continuous data, the quantity has to be smooth;
- simple mathematical derivation for computation ease.

The inverse distance weight function $w^{\mathbf{y}} = \frac{1}{\|y_j - \mathbf{x}_{c_m}\|}$, despite its accuracy and the fact that it appears to be a rational choice is likely to generate high variations potentially leading to discontinuities if $N \ll M$. For the sake of robustness, we instead chose to use the gaussian kernel function $\boldsymbol{G}$, again despite the fact it causes small accuracy discrepancy. It amounts to determine a range of interpolation depending on the particles to grid elements distance. In addition, this weight function turns out to be powerful if the particle density is used for the calculation of its standard deviation at each position $\mathbf{x}_{c_m}$. To illustrate this, we present in Fig. 6 an example of the evaluation of $\mathbf{u}_{\mathcal{C}}$ comparing this method to a fixed standard deviation, showing how the density-based method provides a more robust evaluation for the natural interpretation that can be made of the underlying continuous quantity. However, with such Gaussian kernel, the derivation $w^{\partial \mathbf{y}}$ unfortunately gets a little more complicated.

4.6. Differenciation

For particle control purposes, we will be led to derive equations by some vectorial vertex data noted $\mathbf{f} \in \mathbb{R}^{\dim \times N}$. Eq. 16 shows how the adjacency can be used to estimate new vertex quantities. To help differenciate graph equations, we slightly modify this notation by setting the multiplied vector inside the adjacency matrix, therefore yielding a new weighting type. Considering $\boldsymbol{\psi}$ and $\mathbf{d} \in \mathbb{R}^N$ as scalar data, we write:

$$\boldsymbol{\psi} = \boldsymbol{A}_{w^{\mathbf{f}}}\mathbf{d} = \boldsymbol{A}_{w^{\mathbf{f}}.\mathbf{d}}\vec{1}, \tag{24}$$

where the functional matrix is defined such as: $\boldsymbol{A}_{w^{\mathbf{f}}.\mathbf{d}} = \left[w^{\mathbf{f}}_{ij}d_j\right]$. We will denote the weight derivation $w^{\partial \mathbf{f}}_{ij} = \frac{\partial w^{\mathbf{f}}_{ij}}{\partial f_j} = -\frac{\partial w^{\mathbf{f}}_{ij}}{\partial f_i} \in \mathbb{W}^{\dim}$.

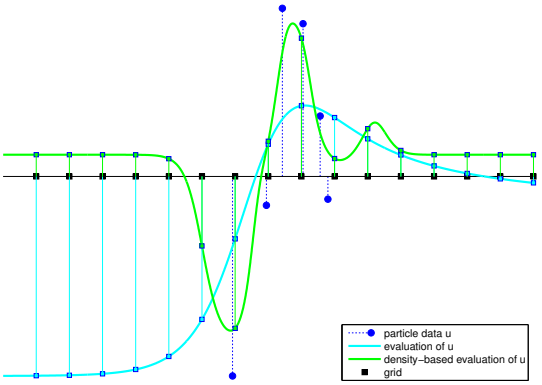


Figure 6. **Evaluation example using particles density.** Evaluation of $\mathbf{u}$ on grid $\mathcal{C}$ obtained by $\mathcal{D}^{-1}_{\boldsymbol{G}^{\mathbf{y}}}\mathcal{A}_{\boldsymbol{G}^{\mathbf{y}}}\mathbf{u}$. Here $N = 6$, $M = 16$, $o = M$. The standard deviation of $\boldsymbol{G}$ is set as the size of the meshes of $\mathcal{C}$ for the simple evaluation, and as the inverse of two times the particle density ($2\mathcal{A}_{\boldsymbol{G}^{\mathbf{y}}}\vec{1}$) for the density-based evaluation.

The derivation $\frac{\partial \boldsymbol{\psi}}{\partial \mathbf{f}} \in \mathbb{W}^{d\times N\times N}$ reads [†]:

$$\frac{\partial \boldsymbol{\psi}}{\partial \mathbf{f}} = \partial_{\mathbf{f}} \boldsymbol{A}_{w^{\mathbf{f}}}\mathbf{d} \tag{25}$$

$$= \partial_{\mathbf{f}} \begin{bmatrix} \sum_j^N w_{1j}^{\mathbf{f}} d_j \\ \sum_j^N w_{2j}^{\mathbf{f}} d_j \\ \dots \\ \sum_j^N w_{Nj}^{\mathbf{f}} d_j \end{bmatrix} \tag{26}$$

$$= \begin{bmatrix} \partial_{f_1} \sum_j^N w_{1j}^{\mathbf{f}} d_j & \partial_{f_2} \sum_j^N w_{1j}^{\mathbf{f}} d_j & \dots & \partial_{f_N} \sum_j^N w_{1j}^{\mathbf{f}} d_j \\ \partial_{f_1} \sum_j^N w_{2j}^{\mathbf{f}} d_j & \partial_{f_2} \sum_j^N w_{2j}^{\mathbf{f}} d_j & \dots & \partial_{f_N} \sum_j^N w_{2j}^{\mathbf{f}} d_j \\ \dots & \dots & \dots & \dots \\ \partial_{f_1} \sum_j^N w_{Nj}^{\mathbf{f}} d_j & \partial_{f_2} \sum_j^N w_{Nj}^{\mathbf{f}} d_j & \dots & \partial_{f_N} \sum_j^N w_{Nj}^{\mathbf{f}} d_j \end{bmatrix} \tag{27}$$

$$= \begin{bmatrix} -\sum_j^N w_{1j}^{\partial \mathbf{f}} d_j & w_{12}^{\partial \mathbf{f}} d_2 & \dots & w_{1N}^{\partial \mathbf{f}} d_N \\ w_{21}^{\partial \mathbf{f}} d_1 & -\sum_j^N w_{2j}^{\partial \mathbf{f}} d_j & \dots & w_{2N}^{\partial \mathbf{f}} d_N \\ \dots & \dots & \dots & \dots \\ w_{N1}^{\partial \mathbf{f}} d_1 & w_{N2}^{\partial \mathbf{f}} d_2 & \dots & -\sum_j^N w_{Nj}^{\partial \mathbf{f}} d_j \end{bmatrix} \tag{28}$$

$$= -\left[\left(\sum_j^N w_{ij}^{\partial \mathbf{f}} d_j\right)\delta_{ik}\right] + \left[w_{ij}^{\partial \mathbf{f}} d_j\right]. \tag{29}$$

One can identify the degree and adjacency matrix out of this expression, leading to the equation:

$$\frac{\partial \boldsymbol{\psi}}{\partial \mathbf{f}} = -\boldsymbol{D}_{w^{\partial \mathbf{f}}\cdot\mathbf{d}} + \boldsymbol{A}_{w^{\partial \mathbf{f}}\cdot\mathbf{d}}. \tag{30}$$

Using the graph Laplacian $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$, we finally obtain:

$$\frac{\partial \boldsymbol{\psi}}{\partial \mathbf{f}} = -\boldsymbol{L}_{w^{\partial \mathbf{f}}\cdot\mathbf{d}}. \tag{31}$$

It becomes also possible to simply write swarm PDEs describing the behavior of particles. In the case $\mathbf{d} = \vec{1}$, one can notice that $\boldsymbol{L}_{w^{\partial \mathbf{f}}}$ is symmetric.

[†] Considering $\boldsymbol{A}$ independent of $\mathbf{f}$, which is an approximation in case $\mathbf{f} = \mathbf{y}$ as seen in Eq. 13, where it depends on the mesh belonging matrix $\mathcal{P}$.

The derivation of Eulerian quantities $\psi_{\mathcal{C}}$ is different since the weight function depends in part on the grid meshes instead of double vertex dependency for classical graph weights. Therefore, the degree matrix does not appear and the derivation of the convolution of $\mathbf{d}$ on $\mathcal{C}$ simply reads:

$$\frac{\partial \psi_{\mathcal{C}}}{\partial \mathbf{f}} = \mathcal{A}_{w^{\partial \mathbf{f}}} \underline{\mathbf{d}}. \tag{32}$$

5. OPTIMAL CONTROL

The evolution model $\mathbb{M}$ of Eq. 1 aims at giving a first assumption of the particle behavior. The evolution of the system state $\mathbf{X}(t)$ can be directly inferred from this model by integrating it over time. The assimilation process is in charge of modifying this state trajectory by taking into account setpoints (or observations) at given times. As a result, the control is defined as a tradeoff between what can be expected from the model and what is actually required by external constraints $\mathbf{Y}(t)$. In fact, the degree of freedom allowing to modify the evolution of the particles predicted by $\mathbb{M}$ is defined by the levels of confidence in the model, the setpoints, and the initial state conditions. In the assimilation process, these levels are related to the covariance matrices. This is explained thoroughly in the next section.

5.1. Problem statement

Assuming the swarm being coarsely driven by the evolution model $\mathbb{M}$, one can add an unknown additive control variable $\boldsymbol{\epsilon}_{\mathbb{M}}$, relative to the noise on the evolution, leading to:

$$\frac{\partial \mathbf{X}}{\partial t} + \mathbb{M}(\mathbf{X}) = \boldsymbol{\epsilon}_{\mathbb{M}}. \tag{33}$$

The setpoint data $\mathbf{Y}(t)$ have to be compared to the model output $\mathbf{X}(t)$, as seen in Fig. 2. This comparison can be direct or not, for example through higher level constraints such as differential particles configuration or local density (as seen in Fig. 1). The relation from such data to $\mathbf{X}$ is therefore not direct, and leads to introducing an observation operator $\mathbb{H}$, formalizing:

$$\mathbb{H}(\mathbf{X}) = \mathbf{Y} + \boldsymbol{\epsilon}_{\mathbb{H}}, \tag{34}$$

where $\boldsymbol{\epsilon}_{\mathbb{H}}$ stands for an error on the setpoint. Note that a direct observation corresponds to $\mathbb{H}$ equals the identity.

Finally, as for the initial condition, we also assume its value to be uncertain, leading to:

$$\mathbf{X}(t_0) = \mathbf{X}_0 + \boldsymbol{\epsilon}_0. \tag{35}$$

Therefore, the problem consists in extracting $\mathbf{X}$ that satisfies the system of relations 33, 34 and 35. This is the *control* issue: how to extract a solution of lower energy on variables $\boldsymbol{\epsilon}_{\mathbb{M}}$ and $\boldsymbol{\epsilon}_0$ (*i.e.* abiding by as much as possible the natural model $\mathbb{M}$ and the initial condition $\mathbf{X}_0$) such that the computed state along time fits the actual setpoints $\mathbf{Y}$, up to the authorized error. This leads to extracting $\mathbf{X}(t)$ that corresponds to the lowest discrepancy between the external constraints and the original trajectory in state space. These conditions can be expressed through the minimization of the cost function $\mathcal{J}$ defined as:

$$\begin{aligned} \mathcal{J}(\boldsymbol{\epsilon}_{\mathbb{M}}, \boldsymbol{\epsilon}_0) = \quad & \frac{1}{2}\int_{t_0}^{t_f} \|\boldsymbol{\epsilon}_{\mathbb{M}}\|^2_{Q^{-1}} dt + \frac{1}{2}\int_{t_0}^{t_f} \|\boldsymbol{\epsilon}_0\|^2_{B^{-1}} dt \\ & + \frac{1}{2}\int_{t_0}^{t_f} \|\mathbf{Y} - \mathbb{H}(\mathbf{X})\|^2_{R^{-1}} dt. \end{aligned} \tag{36}$$

where Q, B and R are respectively the error covariance matrices associated to noises $\boldsymbol{\epsilon}_{\mathbb{M}}$, $\boldsymbol{\epsilon}_0$ and $\boldsymbol{\epsilon}_{\mathbb{H}}$. The expression $\|.\|_{C^{-1}}$ stands for the induced norm of the inner product $\left\langle C^{-1}.,.\right\rangle$ where C is an endomorphism.

These covariances are of high importance in the control process, in the sense that they parameterize the trade-off between the deterministic model and the setpoints. For example, the initial configuration of the swarm at t_0 can be set as fully trusted, and so not modifiable, by parameterizing the associated error to zero and therefore associating an error covariance matrix such that: $B^{-1} = +\infty$. Similar behaviors hold for matrices Q and R.

From a mathematical point of view, in practice, the optimization of Eq. 36 leads to some difficulties in reason of the potentially large size of the system state and to the complexity of the evolution model. A solution to face this problem is to use an adjoint formulation, as presented in the next section.

5.2. Adjoint formulation

As mentioned, the estimation of $\delta\mathcal{J}(\delta\boldsymbol{\epsilon}_0, \delta\boldsymbol{\epsilon}_{\mathbb{M}})$ is in practice unfeasible for a large system's state (here $\mathbf{X} \in \mathbb{R}^{2\dim N}$). As a matter of fact, such an evaluation would require computing perturbations of the state variables along all the components of the control variables $(\delta\boldsymbol{\epsilon}_{\mathbb{M}}, \delta\boldsymbol{\epsilon}_0)$ and all along the temporal grid – i.e. integrating our evolution model for all perturbed components of the control variables which is computationally completely unrealistic. An elegant solution of this problem consists in relying on an *adjoint formulation* [21] by adding an adjoint variable $\boldsymbol{\lambda}$ to the derived model $\partial\mathbb{M}$. With the help of a few expansions and properties, and by defining the linearisation $\partial_{\mathbf{X}}\mathbb{M} \in \mathbb{R}^{2\dim N \times 2\dim N}$ of operator $\mathbb{M}$ (also known as Gâteaux derivative) by:

$$(\partial_{\mathbf{X}}\mathbb{M})d\mathbf{X} = \lim_{\beta\to 0} \frac{\mathbb{M}(\mathbf{X} + \beta d\mathbf{X}) - \mathbb{M}(\mathbf{X})}{\beta}, \tag{37}$$

and the adjoint linearised operator $(\partial_{\mathbf{X}}\mathbb{M})^* \in \mathbb{R}^{2\dim N \times 2\dim N}$ such as:

$$\langle (\partial_{\mathbf{X}}\mathbb{M})d\mathbf{X}, \boldsymbol{\lambda}\rangle = \left\langle d\mathbf{X}, (\partial_{\mathbf{X}}\mathbb{M})^* \boldsymbol{\lambda}\right\rangle, \tag{38}$$

it can be demonstrated that solving the assimilation problem in an incremental framework consists in performing the following algorithm 1 on the basis of the solution $\mathbf{X}(t)$ given by the model integration.

Algorithm 1: Incremental variational data assimilation

Data: Trajectory $\mathbf{X}(t)$
Result: Optimal solution

- Perform integration of $\mathbb{M}$ with initial condition $\mathbf{X}(t_0) = \mathbf{X}_0$

while *No convergence* **do**

- Set $\boldsymbol{\lambda}(t_f) = 0$
- Backward integration of the adjoint linearised model:

$$\frac{\partial\boldsymbol{\lambda}}{\partial t} + (\partial_{\mathbf{X}}\mathbb{M})^* \boldsymbol{\lambda} = (\partial_{\mathbf{X}}\mathbb{H})^* R^{-1} (\mathbf{Y} - \mathbb{H}(\mathbf{X})) \tag{39}$$

- From adjoint trajectory $\boldsymbol{\lambda}(t; \boldsymbol{\epsilon}_{\mathbb{H}})$:
  - compute: $d\mathbf{X}(t_0) = d\boldsymbol{\epsilon}_0 = \frac{1}{t_f - t_0} B\boldsymbol{\lambda}(t_0)$
  - compute: $d\boldsymbol{\epsilon}_{\mathbb{M}}(t) = Q\boldsymbol{\lambda}(t)$
- Forward integration of the linearised model:

$$\frac{\partial d\mathbf{X}}{\partial t} + (\partial_{\mathbf{X}}\mathbb{M})d\mathbf{X} = d\boldsymbol{\epsilon}_{\mathbb{M}} \tag{40}$$

- Update trajectory $\mathbf{X}(t)$ by adding computed $d\mathbf{X}(t)$

end

The method first consists in obtaining the state trajectory $\mathbf{X}(t)$ abiding by the evolution model $\mathbb{M}$ with respect to the given initial condition $\mathbf{X}_0$.

At this stage of the process, the assimilation controller of Fig. 2, and detailed in the above Fig. 7 *(left)*, is in charge of computing the control variables of the system.

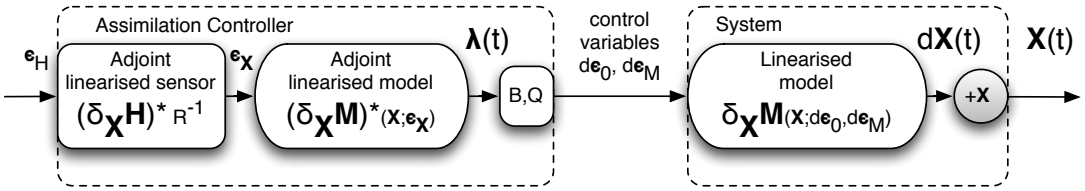


Figure 7. **Details of the assimilation controller and the system of Fig. 2 in an incremental framework.**

Around the state trajectory, the backward integration of the adjoint linearised model of Eq. 39 is in charge of computing $\boldsymbol{\lambda}$ which can be seen as an error variable along time. The evolution of $\boldsymbol{\lambda}$ is in fact:

- powered by a discrepancy measure between the setpoint $\mathbf{Y}$ and the state observation $\mathbb{H}(\mathbf{X})$;
- and guided by the adjoint linearised model $(\partial_{\mathbf{X}}\mathbb{M})^*$.

The variations of the control variables $\boldsymbol{\epsilon}_0$ and $\boldsymbol{\epsilon}_{\mathbb{M}}(t)$ are then calculated using $\boldsymbol{\lambda}$, thus allowing to compute the variation of the state trajectory $d\mathbf{X}(t)$ by the forward integration of Eq. 40 as shown in Fig. 7 *(right)*. Finally, the state correction is added to last obtained state trajectory $\mathbf{X}(t)$, and the overall process is iteratively repeated until convergence.

It is important to notice that both the linearized model and its adjoint evolve around the updated state. To start the control we then must dispose of a first assumption of $\mathbf{X}(t)$, which is done thanks to the direct integration of $\mathbb{M}(\mathbf{X})$, and to an initial condition $\mathbf{X}_0$.

The details of this derived models are now presented, and will be followed by several sensors presentation along with their linearisation.

5.3. Linearized model

The model being defined by the equation system 17, their expression therefore read:

$$\partial_{\mathbf{X}}\mathbb{M} = \begin{bmatrix} 1 & 0 \\ 0 & -\underline{\mathbf{m}}^{-1} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ \partial_{\mathbf{y}}\mathbb{F} & \partial_{\mathbf{u}}\mathbb{F} \end{bmatrix}, \tag{41}$$

$$(\partial_{\mathbf{X}}\mathbb{M})^* = \begin{bmatrix} 0 & \partial_{\mathbf{y}}\mathbb{F}^T \\ 1 & \partial_{\mathbf{u}}\mathbb{F}^T \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -\underline{\mathbf{m}}^{-1} \end{bmatrix}. \tag{42}$$

One can observe that the task consists in correctly expressing the linearization and the transposed linearization of forces $\mathbb{F}$. Thanks to Eq. 17b and Eq. 31, we get:

$$\partial_{\mathbf{y}}\mathbb{F} = -\boldsymbol{L}_{\boldsymbol{r}^{\partial_{\mathbf{y}}}}, \tag{43}$$

$$\partial_{\mathbf{u}}\mathbb{F} = -\underline{\mathbf{k}}. \tag{44}$$

Since $\boldsymbol{r}_{ij}^{\mathbf{y}}$ is a vector, its derivative is a matrix ($\in \mathbb{R}^{d\times d}$). Considering the distance between two particles $D_{ij}^{\mathbf{y}} = \|y_j - y_i\|$, the repulsion force is of form $f(D_{ij}^{\mathbf{y}})e_{ij}^{\mathbf{y}}$ with $f : \mathbb{R} \to \mathbb{R}$. The derivation for any distance-based interaction force, including $\boldsymbol{r}_{ij}^{\mathbf{y}}$, then reads:

$$\boldsymbol{r}_{ij}^{\partial\mathbf{y}} = \frac{1}{D_{ij}^{\mathbf{y}}} f(D_{ij}^{\mathbf{y}}) \left[e_{ij}^{\mathbf{y}}\right]_{\left(\substack{\text{orthoradial}\\ \text{projection}}\right)} - f'(D_{ij}^{\mathbf{y}}) \left[e_{ij}^{\mathbf{y}}\right]_{\left(\substack{\text{radial}\\ \text{projection}}\right)}, \tag{45}$$

which is a symmetric matrix. In the end, transposals of the above linearisations simply give the adjoint linearised terms by reading

$$(\partial_{\mathbf{y}}\mathbb{F})^T = \partial_{\mathbf{y}}\mathbb{F}, \tag{46}$$

$$(\partial_{\mathbf{u}}\mathbb{F})^T = \partial_{\mathbf{u}}\mathbb{F}. \tag{47}$$

5.4. Observation operators

One easily understands that the sensor of Fig. 2 transforms the state $\mathbf{X}$ to a data lying in the same domain as $\mathbf{Y}$. Therefore any user defined setpoint must be associated to an observation operator $\mathbb{H}(\mathbf{X})$. If the setpoint is directly connected to the system's state, this operator is trivial. For example in our case, a full state setpoint reads:

$$\mathbf{Y} = \begin{bmatrix} \mathbf{y}_{sp} \\ \mathbf{u}_{sp} \end{bmatrix} \quad \Rightarrow \quad \mathbb{H}(\mathbf{X}) = \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \end{bmatrix} \quad \Rightarrow \quad (\partial_{\mathbf{X}}\mathbb{H})^* = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{48}$$

In case of higher level observations, the expression of $\mathbb{H}$ and its linearisation can take various forms. In the following we present some observation operators that appear to be useful in the context of particle swarm control.

5.4.1. Lagrangian setpoints In addition to the trivial Lagrangian observation lately presented, we propose two operators based on inter particles relations.

The first operator consists in imposing specific relationships between particles. When such relationships correspond to distances, this observation operator describes shapes [11], but we note that other shape-description possibilities could have been used (see for example [32]). The relation between a single particle p_k and others is defined through a weight function w and a state quantity $\mathbf{f}$. Assuming the relation between particles defined through a matrix $\boldsymbol{M}_{sp} \in \mathbb{W}^{N\times N}$, we can extract the vector of relations of the k-th particle, the desired spatial relations, by writing:

$$\overset{k}{\mathbf{Y}} = \boldsymbol{M}_{sp}\delta^i_k \in \mathbb{W}^N, \tag{49}$$

with $\delta^i_k \in \mathbb{Z}^N$. This setpoint has to be compared to the state by the sensor and through the adjacency matrix defined by weighted connections w, and one of the state's quantities $\mathbf{f}$ belonging to $\mathbf{X}$, leading to:

$$\overset{k}{\mathbb{H}}(\mathbf{X}) = \boldsymbol{A}_{w^{\mathbf{f}}}\delta^i_k. \tag{50}$$

In the rest of the paper we call this a *k-specific adjacency* operator (KSA) associated to the k-specific vector of adjacencies: $\overset{k}{\boldsymbol{a}}_{w^{\mathbf{f}}} \in \mathbb{W}^N$. In order to derive the observation operator, we also introduce the k-specific Laplacian matrix:

$$\overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}}} = \frac{\partial \overset{k}{\boldsymbol{a}}_{w^{\mathbf{f}}}}{\partial \mathbf{f}}, \tag{51}$$

which can be seen as a k-decomposition of the Laplacian matrix, *i.e.*:

$$\sum_k \overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}}} = \boldsymbol{L}_{w^{\partial\mathbf{f}}}. \tag{52}$$

This operator reads:

$$\overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}}} = \begin{cases} \overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}},ij} = w^{\partial\mathbf{f}}_{ij} & \text{if } i = j \\ \overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}},ij} = -w^{\partial\mathbf{f}}_{ij} & \text{if } j = k \\ \overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}},ij} = 0 & \text{if } i = j = k \text{ since } \partial_{f_k} w^{\mathbf{f}}_{kk} = 0 \\ \overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}},ij} = 0 & \text{otherwise} \end{cases} \tag{53}$$

For example, if such a setpoint is applied to every particle p_k, the control process will amount to find $\mathbf{f}$ that minimizes $\|\boldsymbol{M}_{sp} - \boldsymbol{A}_{w^{\mathbf{f}}}\|$. Alternatively, in case $w \in \mathbb{R}$ is a distance function between data $\mathbf{f}$ of the nodes of the graph, this operator is invariant to the shape's translation and rotation.

The second operator is quite the same as the last one. It however introduces a degree of freedom by adding a normalization: the scale of the setpoint is now unspecified. Instead of explicitly impose

Table I. **Overview of the three Lagangian observation operators**.

	$\mathbf{Y}$ setpoint	$\mathbb{H}(\mathbf{X})$ observation operator	$(\partial_{\mathbf{X}}\mathbb{H})^*$ adjoint linearised observation operator
state	$\mathbf{X}_{sp}$	$\mathbf{X}$	I_N
k-specific adjacency	$\boldsymbol{M}_{sp}\delta^i_k$	$\overset{k}{\boldsymbol{a}}_{w^{\mathbf{f}}}$	$\left(\overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}}}\right)^T$
k-specific normalized adjacency	$\boldsymbol{M}_{sp}\boldsymbol{D}^{-1}_{\boldsymbol{M}_{sp}}\delta^i_k$	$\left(\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right)^{-1}\overset{k}{\boldsymbol{a}}_{w^{\mathbf{f}}}$	$\left(\overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}}} - \overset{k}{\mathbb{H}}\otimes\partial_{\mathbf{f}}\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right)^T\left(\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right)^{-1}$

relations between particles with $\boldsymbol{M}_{sp}$, we only desire here to respect the proportion of relation. We name this operator *k-specific normalized adjacency* operator (KSNA). With $\boldsymbol{D}_{\boldsymbol{M}_{sp}} \in \mathbb{W}^{N\times N}$ being the degree matrix of $\boldsymbol{M}$, we thus write:

$$\overset{k}{\mathbf{Y}} = \boldsymbol{M}_{sp}\boldsymbol{D}^{-1}_{\boldsymbol{M}_{sp}}\delta^i_k, \quad \text{and} \quad \overset{k}{\mathbb{H}}(\mathbf{X}) = \boldsymbol{A}_{w^{\mathbf{f}}}\boldsymbol{D}^{-1}_{w^{\mathbf{f}}}\delta^i_k. \tag{54}$$

In other words, by noting $\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}} = \delta^k_i\boldsymbol{D}_{w^{\mathbf{f}}}\delta^i_k \in \mathbb{W}$ the k-specific degree, we obtain:

$$\overset{k}{\mathbb{H}}(\mathbf{X}) = \overset{k}{\boldsymbol{a}}_{w^{\mathbf{f}}}\left(\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right)^{-1}, \tag{55}$$

and its derivation gives:

$$\begin{aligned}\partial_{\mathbf{f}}\overset{k}{\mathbb{H}} &= -\left(\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right)^{-2}\left(\overset{k}{\boldsymbol{a}}_{w^{\mathbf{f}}}\otimes\partial_{\mathbf{f}}\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right) + \left(\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right)^{-1}\overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}}},\\ &= \left(\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right)^{-1}\left(\overset{k}{\boldsymbol{l}}_{w^{\partial\mathbf{f}}} - \overset{k}{\mathbb{H}}\otimes\partial_{\mathbf{f}}\overset{k}{\boldsymbol{d}}_{w^{\mathbf{f}}}\right).\end{aligned} \tag{56}$$

The KSA and KSNA operators, alike the direct state operator, are applied to Lagrangian setpoints. This is summarized in Tab. I. We will now turn to the management of environment data into the sensor process, that we refer as Eulerian constraints.

5.4.2. Eulerian setpoints In some specific applications, it is also of great interest to control the swarm by continuous data. For example, one can impose at a given time the concentration of particles to respect a continuous constraint defined on a grid $\mathcal{C}$. For the sake of clarity, let us consider a system driven by the model $\mathbb{M}$ whose properties are $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{q}$ the quantity of particles (the system's state being composed of $\mathbf{y}$, $\mathbf{u}$). To control the system either on the basis of the position, velocity or quantity, we need to define two operators: the **density observation operator** that links the system state to a continuous quantity defined on the grid and the **projection observation operator** that is in charge of estimating the values of the given component on a grid. These two operators are detailed below.

Density observation operator. In this case, we seek to express the concentration of particles $\mathbf{q}$ on the grid. The observation operator seen in Eq. 21 is such as: $\mathbb{H}(\mathbf{X}) = \boldsymbol{\rho}_{\mathcal{C}} = \mathcal{A}_{\boldsymbol{G}^{\mathbf{y}}}\mathbf{q}$, and its derivation by $\mathbf{X}$ simply reads:

$$\partial_{\mathbf{X}}\mathbb{H} = \begin{bmatrix}\partial_{\mathbf{y}}\mathbb{H}\\ \partial_{\mathbf{u}}\mathbb{H}\end{bmatrix} = \begin{bmatrix}\mathcal{A}_{\boldsymbol{G}^{\partial\mathbf{y}}}\underline{\mathbf{q}}\\ 0\end{bmatrix}. \tag{57}$$

Projection observation operator. As mentioned above, its goal is to evaluate on a grid $\mathcal{C}$ some variables. The velocity $\mathbf{u}$ accounts for a rational data to estimate on $\mathcal{C}$, and its projection on it is already given in Eq. 23 such as: $\mathbb{H}(\mathbf{X}) = \mathbf{u}_{\mathcal{C}} = \mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{A}_{w^{\mathbf{y}}}\mathbf{u}$. Its derivation by $\mathbf{X}$ reads:

$$\partial_{\mathbf{X}}\mathbb{H} = \begin{bmatrix}\partial_{\mathbf{y}}\mathbb{H}\\ \partial_{\mathbf{u}}\mathbb{H}\end{bmatrix} = \begin{bmatrix}\mathcal{D}^{-1}_{w^{\mathbf{y}}}\left(\mathcal{A}_{w^{\partial\mathbf{y}}}\underline{\mathbf{u}} - \underline{\mathbf{u}_{\mathcal{C}}}\mathcal{A}_{w^{\partial\mathbf{y}}}\right)\\ \mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{A}_{w^{\mathbf{y}}}\end{bmatrix} \tag{58}$$

Table II. **Eulerian observation operators**.

	$\mathbf{Y}$ setpoint	$\mathbb{H}(\mathbf{X})$ observation operator		$(\partial_{\mathbf{X}}\mathbb{H})^*$ adjoint linearised observation operator	
density	$\boldsymbol{\rho}_{\mathcal{C},sp}$	$\mathcal{A}_{\boldsymbol{G}^{\mathbf{y}}}\mathbf{q}$	$\Rightarrow$ Eq. 21	$\begin{bmatrix}\underline{\mathbf{q}}\mathcal{A}^T_{\boldsymbol{G}^{\partial\mathbf{y}}}\\ 0\end{bmatrix}$	$\Rightarrow$ Eq. 57
velocity	$\mathbf{u}_{\mathcal{C},sp}$	$\mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{A}_{w^{\mathbf{y}}}\mathbf{u}$	$\Rightarrow$ Eq. 23	$\begin{bmatrix}\mathcal{S}_{w^{\partial\mathbf{y}}.\mathbf{u}}\\ \mathcal{A}^T_{w^{\mathbf{y}}}\end{bmatrix}\mathcal{D}^{-1}_{w^{\mathbf{y}}}$	$\Rightarrow$ Eq. 58
divergence	$\boldsymbol{\xi}_{\mathcal{C},sp}$	$\underline{\boldsymbol{\nabla}}\mathbf{u}_{\mathcal{C}}$	$\Rightarrow$ Eq. 59	$\begin{bmatrix}(\partial_{\mathbf{y}}\underline{\boldsymbol{\nabla}}\mathbf{u}_{\mathcal{C}})^T\\ (\partial_{\mathbf{u}}\underline{\boldsymbol{\nabla}}\mathbf{u}_{\mathcal{C}})^T\end{bmatrix}$	$\Rightarrow$ Eq. 61
curl	$\boldsymbol{\zeta}_{\mathcal{C},sp}$	$[\underline{\boldsymbol{\nabla}}\times]\,\mathbf{u}_{\mathcal{C}}$	$\Rightarrow$ Eq. 60	$\begin{bmatrix}(\partial_{\mathbf{y}}\,[\underline{\boldsymbol{\nabla}}\times]\,\mathbf{u}_{\mathcal{C}})^T\\ (\partial_{\mathbf{u}}\,[\underline{\boldsymbol{\nabla}}\times]\,\mathbf{u}_{\mathcal{C}})^T\end{bmatrix}$	$\Rightarrow$ Eq. 62

where for clarity sakes, we have introduced the Eulerian *deviation* operator $\mathcal{S}_{w^{\partial\mathbf{f}}.\mathbf{d}} = \mathcal{A}_{w^{\partial\mathbf{f}}}\underline{\mathbf{d}} - \underline{\mathbf{d}_{\mathcal{C}}}\mathcal{A}_{w^{\partial\mathbf{f}}}$ allowing to expressing derivations in a more compact way.

In addition to these operators inspired from particles concentration and velocity, we propose higher level setpoints based on the derivatives of the velocity. More precisely, we use:

- the **divergence**, which expresses the dynamic of dilatation or concentration of particles:

$$\mathbb{H}(\mathbf{X}) = \boldsymbol{\xi}_{\mathcal{C}} = \underline{\boldsymbol{\nabla}}\mathbf{u}_{\mathcal{C}} = \mathcal{D}^{-1}_{w^{\mathbf{y}}}(\mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{D}_{w^{\partial\mathbf{y}}}\mathcal{A}_{w^{\mathbf{y}}} - \mathcal{A}_{w^{\partial\mathbf{y}}})\mathbf{u}, \tag{59}$$

- and **vorticity** (or rotational or curl), related to the dynamic of whirling of the particles:

$$\mathbb{H}(\mathbf{X}) = \boldsymbol{\zeta}_{\mathcal{C}} = [\underline{\boldsymbol{\nabla}}\times]\,\mathbf{u}_{\mathcal{C}} = \mathcal{D}^{-1}_{w^{\mathbf{y}}}([\underline{\boldsymbol{\nabla}}\times]\,\mathcal{A}_{w^{\mathbf{y}}} + \mathcal{D}^{-1}_{w^{\mathbf{y}}}\,[\mathcal{D}_{w^{\partial\mathbf{y}}}\times]\,\mathcal{A}_{w^{\mathbf{y}}})\mathbf{u}. \tag{60}$$

The result of the derivation of these operators is directly expressed as follows:

- for divergence:

$$\partial_{\mathbf{X}}\mathbb{H} = \begin{bmatrix}\partial_{\mathbf{y}}\mathbb{H}\\ \partial_{\mathbf{u}}\mathbb{H}\end{bmatrix} = \begin{bmatrix}-\mathcal{D}^{-1}_{w^{\mathbf{y}}}\underline{\boldsymbol{\xi}_{\mathcal{C}}}\mathcal{A}_{w^{\partial\mathbf{y}}} + \mathcal{D}^{-2}_{w^{\mathbf{y}}}\mathcal{D}_{w^{\partial\mathbf{y}}}\mathcal{S}_{w^{\partial\mathbf{y}}.\mathbf{u}} - \mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{S}_{w^{\partial^2\mathbf{y}}.\mathbf{u}}\\ \mathcal{D}^{-1}_{w^{\mathbf{y}}}(\mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{D}_{w^{\partial\mathbf{y}}}\mathcal{A}_{w^{\mathbf{y}}} - \mathcal{A}_{w^{\partial\mathbf{y}}})\end{bmatrix}; \tag{61}$$

- for vorticity:

$$\partial_{\mathbf{X}}\mathbb{H} = \begin{bmatrix}\partial_{\mathbf{y}}\mathbb{H}\\ \partial_{\mathbf{u}}\mathbb{H}\end{bmatrix} = \begin{bmatrix}-\mathcal{D}^{-1}_{w^{\mathbf{y}}}\underline{\boldsymbol{\zeta}_{\mathcal{C}}}\mathcal{A}_{w^{\partial\mathbf{y}}} + \mathcal{D}^{-2}_{w^{\mathbf{y}}}\mathcal{D}_{w^{\partial\mathbf{y}}}\underline{\times}\mathcal{S}_{w^{\partial\mathbf{y}}.\mathbf{u}} - \mathcal{D}^{-1}_{w^{\mathbf{y}}}\mathcal{S}_{w^{\partial^2\mathbf{y}}\times\mathbf{u}}\\ \mathcal{D}^{-1}_{w^{\mathbf{y}}}(\mathcal{D}^{-1}_{w^{\mathbf{y}}}\,[\mathcal{D}_{w^{\partial\mathbf{y}}}\times]\,\mathcal{A}_{w^{\mathbf{y}}} + [\underline{\boldsymbol{\nabla}}\times]\,\mathcal{A}_{w^{\mathbf{y}}})\end{bmatrix}. \tag{62}$$

These observation operators, along with their derivation required by the control process, illustrate the variety of possible phenomena relative to a Lagrangian model. The Eulerian ones are summarized in Tab. II. Let us now turn to the experiment results.

6. RESULTS

In this section, we test and validate our control approach by imposing Lagrangian and Eulerian setpoints to a particle swarm submitted to the dynamical model presented in Section 3. Before describing these experiments, we first discuss some practical aspects in the next paragraph.

6.1. General considerations for experiments

Below are some general settings:

- for the ease of presentation, the experiments are performed in 2D space (i.e. $\dim = 2$);

- the simulations of $\mathbb{M}$, $\partial_{\mathbf{X}}\mathbb{M}$, and $(\partial_{\mathbf{X}}\mathbb{M})^*$ are performed using the 4-th order Runge-Kutta time integration to guarantee stable trajectories, and the time step is set to $0.1\,s$;
- we use a grid $\mathcal{C}$ of 32×32 meshes, each one sizing $1m \times 1m$;
- the mass of particles $\mathbf{m}$ is set to $\vec{1}\,kg$ for every experiment.

As for the error covariance matrices:

- as the relation of Eq. 17a between particles positions and velocities is considered as perfect, we set $Q_{\mathbf{y}}$ null. As a consequence, only $Q_{\mathbf{u}}$ related to the error covariance matrix on the dynamic modeling in Eq. 17b will be non null;
- covariances are parameterized for a slow convergence to avoid oscillations and so improve clarity of method's behavior;
- concerning the initial condition, in order to stick with a constrained initial position, we set it null in this paper for both $B_{\mathbf{y}}$ and $B_{\mathbf{u}}$.

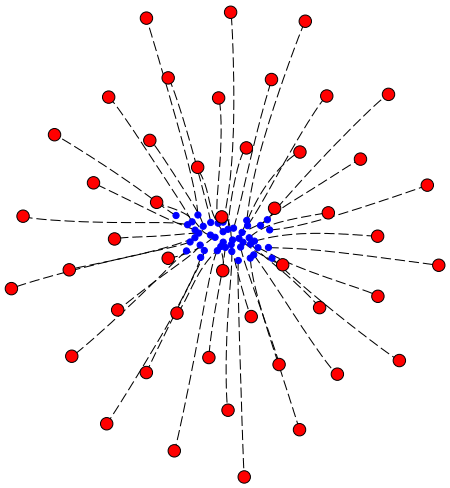
Regarding computation performances, each control loop duration of the examples presented lasted averagely between 10 and 30 seconds on a 2.3 Ghz laptop. Let us also specify that we used the *store all* method consisting of saving the states (positions and velocities) all along the sequence. In case of very big state space, for instance here a large number of particles, and/or very large number of time steps, the needed memory can become significant, which may appear as a drawback of this method, but did not appear as a critical issue in the following experiments. We now turn to experiments with Lagrangian observations. All the experimentations are performed with the parameter values of Tab. III.

6.2. Lagrangian observations

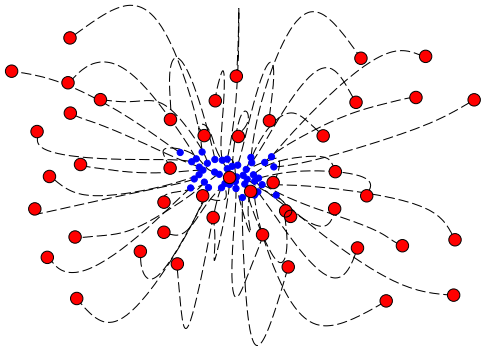
In this example, the goal is to impose a relative relationship between particles. To that end, the KSNA operator detailed in Section 5.4.1 will be used. Although it is formalized for any graph data and any weight function, we will show, for the sake of clarity and in order to have a better understanding of the assimilation process, the results by taking: $\mathbf{f} = \mathbf{y}$ and $w_{ij}^{\mathbf{f}} = D_{ij}^{\mathbf{y}} = \|y_j - y_i\|$, the Euclidean distance function. Considering a spatial configuration of particles (a pattern), we can build its adjacency matrix following Section 5.4 and obtain $\boldsymbol{M}_{sp}$ defining the distance of every particle to all others. The experiment, therefore, consists in bending the model $\mathbb{M}$ to obtain at a given time t_{sp} a spatial configuration of the particles being as close as possible to the targeted pattern with respect to the model $\mathbb{M}$.

We consider an adjacency matrix of distances $\boldsymbol{M}_{sp}$ built thanks to a given swarm pattern, and will apply the KSNA operator for every particle p_k. The setpoint pattern will be the word *TAC* (standing for *Tensorial Adjoint Control*) at $t_{sp} = 12\,s$. It is built offline and constituted of $N = 46$ particles. The initial condition on positions $\mathbf{y}_{t_0}$ is taken as the desired pattern but at a smaller scale as compared to the repulsion magnitude a and the duration lasting up to the setpoint. Therefore, the integration of the evolution model will significantly disperse particles over time. This leaves trajectories of particles which can be used to easily highlight the way the solution converges. So far, we would obtain at t_{sp} a configuration of particles which relative positions do not need to be exchanged. A small randomization around $\mathbf{y}_{t_0}$ is added to break this arrangement and solicit more hardly the repulsive term of the model during the control.

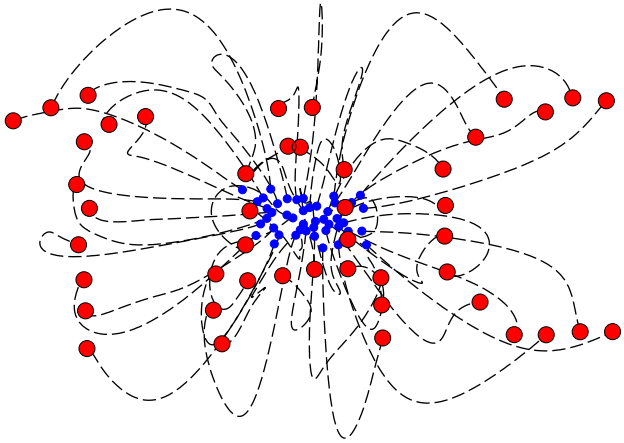
The result of the experiment appears in Fig. 8. The first step (i.e. simulation of the model) in Fig. 8(a) shows that the swarm has no other behavior than diffusion due to particle repulsion. After a few control loops , the swarm changes its whole state quickly and a new behavior appears, as shown on the trajectories of Fig. 8(b). With 200 assimilation iterations (Fig. 8(c)), the word TAC appears clearly from a qualitative point of view. However the swarm begins to encounter difficulties due to particles impediment and the rate of evolution over control loops start to considerably slowdown, as the normalized Root Means Square (RMS) of $\mathbb{H}(\mathbf{X}) - \mathbf{Y}$ shows in Fig. 8(d). Actually, at this stage, the swarm is struggling to overcome the model which does not easily allow for reconfiguration of the swarm, since the initial condition has broken the ideal one as said previously. This is why at convergence, in Fig. 8(c) we can observe particles switching their position by a careful rotation, as



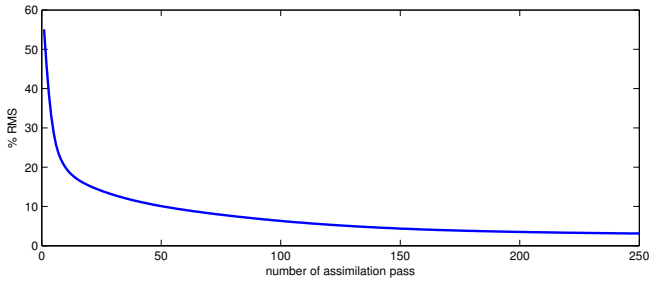
(a) Model integration



(b) Control after 10 assimilation iterations



(c) Control after 200 assimilation iterations



(d) RMS

Figure 8. **KSNA experimentation**. Red dots represent swarm configuration at $t = 12\,s$, blue dots at $t_0 = 0s$. Dashed lines are particle trajectories over sequence duration. Covariances are tuned for a slow and steady convergence to preserve the dynamic of the swarm.

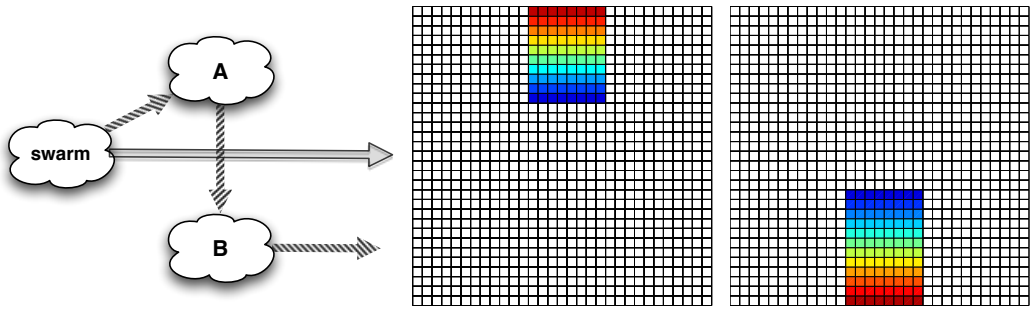
in the left side of the *C* letter. The two closest particles (top of letter *A*) are still trying to switch their position. They are pushed toward each other by the adjoint linearised observation operator $(\partial_{\mathbf{X}}\mathbb{H})^*$, but repulsed by the model. This is a typical local minimum case and drawback of the method These particles are stuck in a bad configuration regarding to the setpoint, and cause the others' positioning to be polluted, since the problem is over constrained when the setpoint pattern $\boldsymbol{M}_{sp}$ is fully connected. Using other shape descriptors by advanced multiresolution strategies or stochastic gradients could for instance lower this problem by minimizing the number of local minima. However this is out of the scope of this paper. Another possibility to overcome this difficulty under the proposed framework would be to decrease the confidence in the model, or increase the setpoint confidence by adjusting the covariance matrices. It is also possible to sparse the latter one in order to relax the control problem.

Last, we obviously notice that scale orientation and positioning of the pattern obtained at convergence correspond to the apparently closest solution in reference to the simulation (a focus on orientation reveals that the obtained pattern is not horizontally oriented) which proves the efficiency of our approach.

6.3. Eulerian observation

In this section, we present experiments with two kinds of Eulerian setpoints based on density and on divergence plus vorticity.

6.3.1. Aggregation density control **Problem statement.** The aggregation problem is well-known when dealing with robot swarms [9, 33]. The problem can be stated as: *"how particles have to move in order to be found in an area at a specific time t_{sp} ?"*. We propose to answer this question by using the density observation operator. The control issue then states how to bend the model $\mathbb{M}$ in order to find a specific concentration of particles at t_{sp}. We will consider the aggregation case illustrated in Fig. 9(a), where the swarm has to be located at area *A* then area *B* while heading East. These locations have been chosen with high inconsistencies regarding the model dynamics in order to highlight the contribution of the control approach. As shown in Fig. 9, we add to these setpoints a density gradient in order to also control the way particles are aggregated.



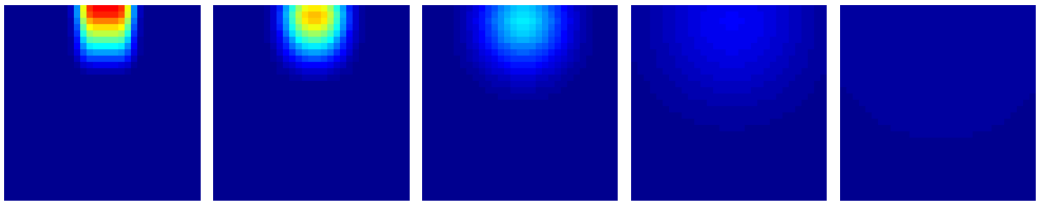
(a) Aggregation problem (b) for area *A*,$t_{sp} = 12\,s$ (c) for area *B*,$t_{sp} = 20\,s$

Figure 9. **Aggregation problem** Instead of being driven only by the evolution model $\mathbb{M}$, we also impose the swarm to reach areas *A* and *B*. The two observations A and B are horizontally symmetric, and maximum value in red reaches $1.3\,ptl.m^{-2}$. Grid $\mathcal{C}$ is only displayed here for further clarity's sake.

The pitch of the grid ($1\,m$) provides the finest relevant standard deviation that we note σ_1. This value is defined as the high resolution and is used for the Gaussian kernel needed by the density observation operator of Eq. 21. One can understand thanks to Fig. 9 that this value is low as compared to the distance between the initial swarm trajectory and aggregation areas. Therefore, the Gaussian derivative driving the density adjoint linearised observation operator (see Tab. II) is of low intensity in the first control iterations, and is not able to provide enough power to the adjoint model (see Eq. 39) in the scope of attracting the swarm to aggregation areas. It will in fact be the

most efficient for particles being at a $1\,m$ distance of aggregation information. Consequently, we suggest embedding the process of sensing with the following technique.

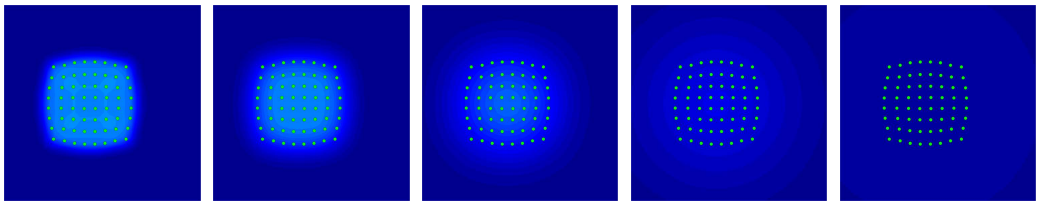
A Multiresolution framework is achieved using a spectrum of standard deviation, instead of the unique usage of σ_1. This allows the particles to capture information at different ranges where the setpoint remains unfulfilled. The spectrum is taken as $\sigma_r = \{1\,m, 2\,m, 4\,m, 8\,m, 16\,m\}$, guaranteeing the distance from a particle p_j to a mesh c_m always varies in the range of two standard deviations: $2\sigma_r < \|\mathbf{y}_j - \mathbf{x}_{c_m}\| < 2\sigma_{r+1}$. We also convolute the setpoint $\mathbf{Y}$ of Fig. 9 for these resolutions in order to provide reachable data to the assimilation process, since the input densities provided for the setpoints are not reachable for a Gaussian kernel convolution, and more widely by any kernel whose derivation is not null. Fig. 10 shows these convolutions.



(a) $\sigma = 1\,m$ (b) $\sigma = 2\,m$ (c) $\sigma = 4\,m$ (d) $\sigma = 8\,m$ (e) $\sigma = 16\,m$

Figure 10. $\mathbf{Y}$ **for different resolutions**. Setpoint for the *A* area at $t_{sp} = 12\,s$

So far, the multiresolution operators and the adjoints of their linearisation correspond to a filtering, whose power is diffused essentially between distances $0\,m$ and $2\sigma_r$. In the same purpose, if we consider the observed density after the integration of model $\mathbb{M}$, shown in Fig. 11, and consider the difference between maps of Figs. 10(e) and 11(e), we observe that the difference $\mathbf{Y} - \mathbb{H}(\mathbf{X})$ is of low intensity, and we easily presume this difference will be much higher for high resolutions once the swarm has reached the area *A* and its neighborhood.



(a) $\sigma = 1\,m$ (b) $\sigma = 2\,m$ (c) $\sigma = 4\,m$ (d) $\sigma = 8\,m$ (e) $\sigma = 16\,m$

Figure 11. $\mathbb{H}(\mathbf{X})$ **for different resolutions right after model integration**.

In fact the covariance R of the setpoint error $\boldsymbol{\epsilon}_{\mathbb{H}}$ strongly depends on the observation operator $\mathbb{H}(\mathbf{X})$. It is meant to amplify the intensity of the adjoint linearised observation operator $(\partial_{\mathbf{X}}\mathbb{H})^*$. Indeed, the confidence in the observation is necessarily higher for low resolutions and we suppose it proportional to the mean density squared[‡]. Therefore we state:

$$R = k_R^2 \boldsymbol{G}_\sigma(0)^2 I_M, \tag{63}$$

with k_R being a user coefficient same for every resolution. We present in Fig. 12 a global behavior of multiresolution adjoint linearised observation operator $(\partial_{\mathbf{X}}\mathbb{H})^*$ amplified by R^{-1} by plotting $\boldsymbol{G}'_\sigma(\mathbf{x})\boldsymbol{G}_\sigma(0)^{-2}$ for the considered standard deviation.

Results. We present in Fig. 13 the results of the experiment by showing the evolution of trajectories of the particles (13(a) - 13(c)) through the control. One can see that the initial trajectories

[‡]We remind the reader that the covariance definition states among others: $[R] = [\mathbb{H}]^2$, with $[.]$ being the quantity dimension operator

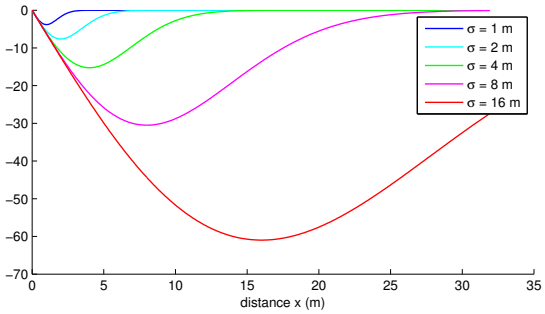


Figure 12. **Behavior of multiresolution density adjoint linearised observation operator $G'_\sigma(\mathbf{x})G_\sigma(0)^{-2}$ amplified by R^{-1}.** Peak of power is reached at σ and intensity grows as resolution decreases

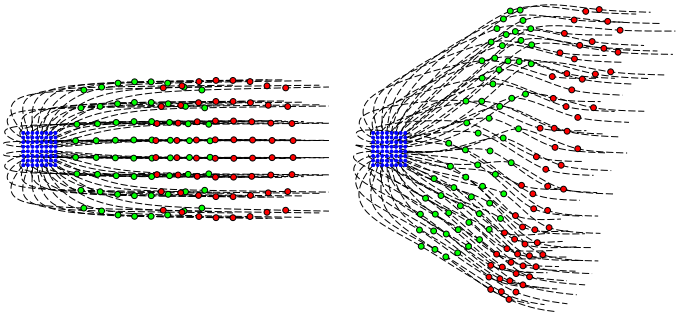
are straight, the swarm has no other purpose than heading East with particles pushing each other back. After 10 assimilation iterations, the trajectories are bended. It is difficult for the swarm to be present at area *A* and only 8 seconds later at area *B*. Therefore, the 'bending' of the model is hard and requires more control loops in order to fulfill reasonably the input densities. After 300 iterations, the swarm has clearly reached the desired areas and the density gradient appears. The particles which were the farthest from the areas managed to get a little closer to *A* and *B* but did not reach it since $\mathbf{Y}$ is almost fulfilled.The error is by then too low to bend their trajectories enough. This is the concern of the lowest resolution adjoint observation operators, and we see in Figs. 13(d) to 13(e) that the observations are reached faster at these resolutions than at the highest one. In Figs. 13(f) to 13(j) it can be seen that the densities observed at near convergence are very close to the demanded setpoints in Figs. 10(a) to 10(e). In addition, it is remarkable that the dynamic has been preserved as much as possible: particles still repulse each other, even if the swarm is asked to be more concentrated than usual.

This experiment demonstrates the ability of the approach to generate consistent trajectories with specific constraints on densities. Let us now turn to constraints on divergence and vorticity.

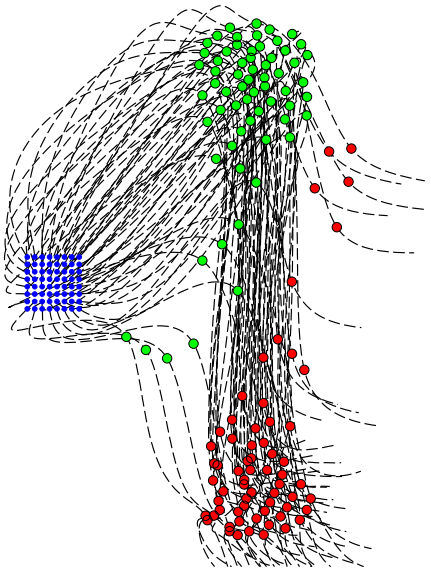
6.3.2. Divergence and vorticity control In addition to the aggregation problem, we propose Eulerian setpoint of even higher level based on the divergence and vorticity. As mentioned above, these quantities are specially interesting to study since they embed key dynamic properties. Moreover, producing a vector field starting from divergence and vorticity maps is not immediate since it requires solving Poisson equations (for instance with the help of the Fourier transform) which is a tricky task. Producing consistent trajectories where specific values of divergence and vorticity are targeted, without explicitly computing the motion field, is then a very appealing and original task that we suggest to solve by using the assimilation process.

In this example we seek to create a swarm dipole made of symmetric homogeneous vorticity and null divergence. Fig. 14(a,b) shows an overview of the control.

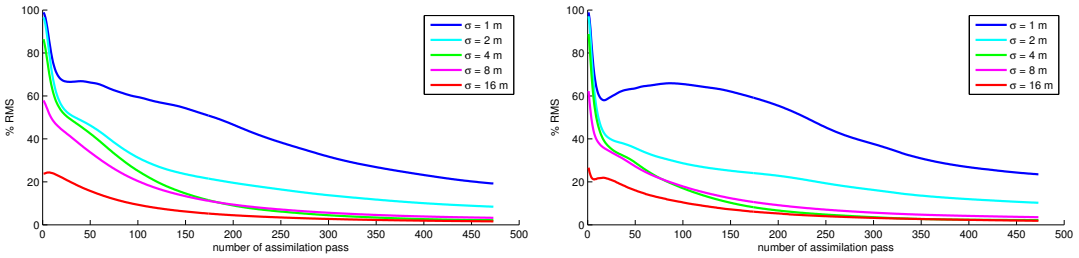
For the experiment, we use the adaptive density based evaluation presented in part 4.5. We present in Fig. 14 the results of the experiment by showing again trajectories of the particles, the evolution of error $\epsilon_{\mathbb{H}}$ and the observed Eulerian values. Fig. 14(b) shows the setpoint $\mathbf{Y}$ for vorticity which is not homogeneous. Indeed, the kernel function used for estimation being the Gaussian function, the setpoint has to be pre-smoothed (as for density observation) to stick with an homogeneous Lagrangian vorticity. The divergence setpoint, being null, is located in the same area as vorticity. The observed vorticity $\mathbb{H}(\mathbf{X})$ (Fig. 14(g)) after 150 assimilation iterations clearly stick to the setpoint $\mathbf{Y}$ (Fig. 14(b)). Divergence is also maintained to a low value as showed in Fig. 14(h). The error on vorticity exponentially decreases (Fig. 14(c)), except in the early iterations while the swarm is not located yet in the setpoint area. One can also notice this experiment has a behavior similar to the aggregation experiment. The swarm looks attracted to the observation area, without any other specification than vorticity and divergence. Actually, the less costly way for particles to create vorticity somewhere is to have a low velocity in the demanded area, instead of high velocity far



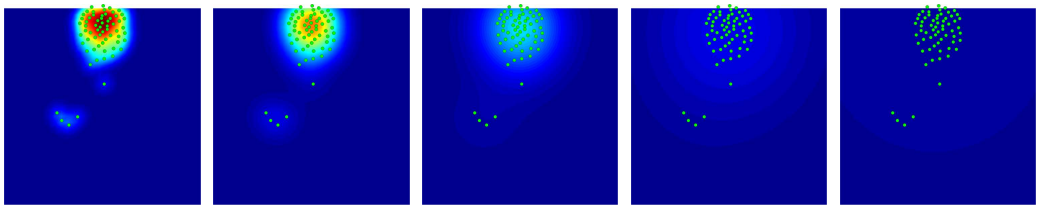
(a) Model Integration (b) Control after 10 assimilation iterations



(c) Control after 300 assimilation iterations



(d) RMS for $t_{sp} = 12\,s$ (e) RMS for $t_{sp} = 20\,s$



(f) $\sigma = 1\,m$ (g) $\sigma = 2\,m$ (h) $\sigma = 4\,m$ (i) $\sigma = 8\,m$ (j) $\sigma = 16\,m$

Figure 13. **Aggregation experimentation**. Green dots represent configuration at $t = 12\,s$, red dots at $t = 20\,s$, and blues at $t_0 = 0\,s$. Figs. 13(f) to 13(j) show $\mathbb{H}(\mathbf{X})$ after 470 assimilation iterations.

from it. Lastly, it is of prime importance to note that the axis of vorticity (low velocity value in Fig. 14(i)) is not located in the middle of each sub-swarm, which would cause the average velocity to be null. Instead, the axis is naturally set on the periphery to maintain the initial average velocity of the swarm ($1\,m.s^{-1}$ East).

The initial and targeted divergence being both very low, the evolution of the error along iterations is not relevant, and is essentially consequence of vorticity control.

7. CONCLUSION AND DISCUSSION

In this paper we have proposed a complete framework to deal with the control of dynamic particle swarm using the variational assimilation theory. This theoretical framework is able to manage both an evolution model and setpoints related to the environment, making it possible to produce complex swarm behaviors while conserving the particle dynamics.

As the usual data assimilation formalism can become heavy in a Lagrangian space, we have proposed an enhanced graph formalism to deal with numerous data, operators and spaces in a unified, compact and generic way. This last part is in itself, to our opinion, a key contribution since it opens new convenient ways to deal with Lagrangian-Eulerian interactions.

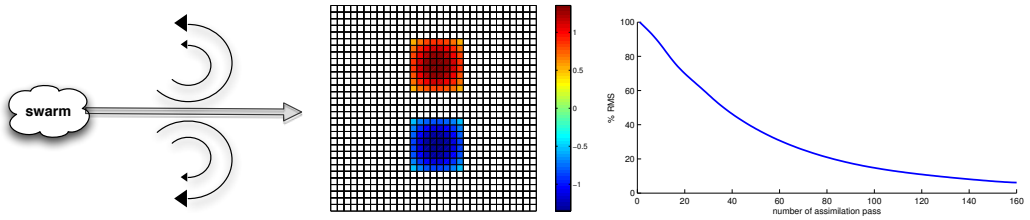
The proposed setpoints in this paper are related to common issues essentially based on spatial configuration like formation or aggregation of the swarm. We have also introduced higher level observation operators like velocity divergence and vorticity that are defined in an Eulerian space in order to demonstrate the ability of the method to deal with highly complex demands.

The variety of experimented observation operators, and the ability of the method to reach the associated setpoints while still preserving as much as possible the particle dynamic, suggests that a large spectrum of other swarm behavior can be reached. This paper also proposes insights on the management of potentially tricky operators like density, by introducing a multiresolution approach. A reflexion on covariance has also been conducted, transposable to other types of operators that were not considered in this paper.

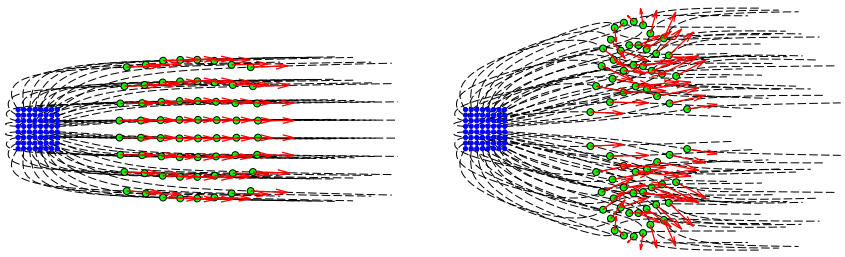
If the particle swarm control by assimilation has proved to be powerful, there exists nonetheless some drawbacks. The first one comes from the assimilation theory which is based on gradient descent. The control is therefore strongly exposed to local minima and we have no assurance to extracting an optimal solution. This is specially true in the Lagrangian space as compared to the Eulerian initial background of assimilation, where particle interaction can impede each other and create energy walls impossible to overcome. The presented results also suggest that the Lagrangian control essentially acts as bending dynamics if referring to the evolution of trajectories along the control loops. Indeed, one can consider them as thin rigid material on which setpoints act as external forces, and particles interactions as internal forces. In addition, the minimization rate depends on covariances which can be tricky to parameterize. For instance, it is hard to compare the model error's covariance with those of the setpoints errors which do not lie in the same quantity space. As a consequence, applying the control privileging one of them with a meaningful factor and with a given minimization speed is almost unfeasible.

In the scope of future work, this approach can be extended to diverse situations and domains. For example, a derivable environment potential can simply be added to the model to guide the swarm instead of having a constant source term. In a larger scope, the state can be composed of more abstract data as compared to position and velocity, and the model can be more advanced than a simple drive-repulse evolution. It can also be considered as perfect, the control thus providing an estimation of unknown model parameters using various real world observations in place of user-defined setpoints.

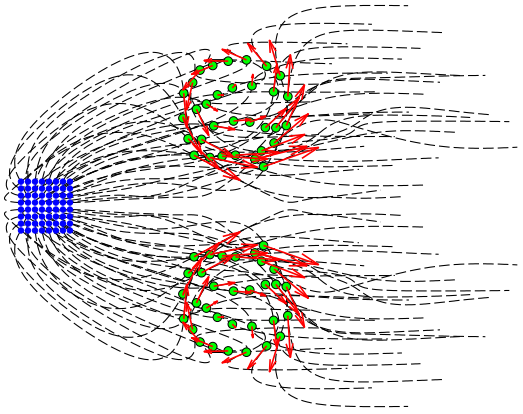
As for the different applications, our approach can be applied to diverse situations where the management of graph data is of high importance. One can for instance mention the environmental sciences (meteorology or oceanography) where it is sometimes useful to represent a scalar quantity with Lagrangian *particles* that correspond to a key event (like a vortex). In that case, as the underlying dynamics is continuous and as some local observations are available with dedicated probes, the proposed framework is well adapted. Lastly, the presented technique dealing with



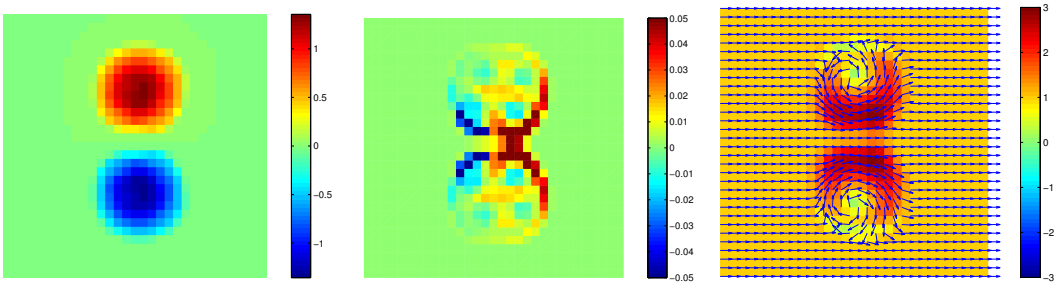
(a) Dipole problem (b) Vorticity setpoint, s^{-1} (c) RMS for $t_{sp} = 14\,s$



(d) Model integration (e) Control after 20 assimilation iterations



(f) Control after 150 assimilation iterations



(g) Vorticity, s^{-1} (h) Divergence, s^{-1} (i) Velocity norm, $m.s^{-1}$

Figure 14. **Dipole problem**. Instead of being driven only by the evolution model $\mathbb{M}$, we also want the swarm to create a velocity dipole by only specifying vorticity and a null divergence. Green dots represent configuration at $t = 14\,s$, and blue dots at $t_0 = 0\,s$. Fig. 14(g) and 14(h) show $\mathbb{H}(\mathbf{X})$ after 150 assimilation iterations. The velocity norm is also presented in Fig. 14(i).

Lagrangian/Eulerian issues can be embedded in other assimilation systems than the variational one, as for instance the particle or ensemble Kalman filters, to overcome the difficulties related to the local minima mentioned above.

ACKNOWLEDGEMENTS

This work has been supported by french *Région Bretagne* by contract reference *RB ARE09143-ARED ASFOULE*.

REFERENCES

1. Schweitzer F. *Brownian Agents and Active Particles: Collective Dynamics in the Natural and Social Sciences*. Springer Berlin Heidelberg, 2007.
2. Huepe C, Zschaler G, Do AL, Gross T. Adaptive-network models of swarm dynamics. *New Journal of Physics* 2011; **13**(7):073 022.
3. Feddema J, Schoenwald D. Decentralized control of cooperative robotic vehicles: Theory and applications. *IEEE Trans. on Robotics and Automation* 2002; **18**(5):852–864.
4. Parker C, Zhang H. Collective unary decision-making by decentralized multiple-robot systems applied to the task-sequencing problem. *Swarm Intelligence* 2010; **4**:199–220.
5. Reynolds CW. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.* August 1987; **21**:25–34.
6. Helbing D, Farkas I, Vicsek T. Simulating dynamical features of escape panic. *Nature* 2000; **407**(1):487–490.
7. Olfati-saber R, Murray M. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. on Automatic Control* 2004; **49**(9):1520–1533.
8. Antonelli G, Arrichiello F, Chiaverini S. Flocking for multi-robot systems via the null-space-based behavioral control. *Swarm Intelligence* 2010; **4**:37–56.
9. Gazi V, Fidan B, Hanay YS, Koksal MI. Aggregation, foraging, and formation control of swarms with non-holonomic agents using potential functions and sliding mode techniques. *Turkish J. Elect. Eng. Comput. Sci.* July 2007; **15**(2):149–168.
10. Turgut A, Çelikkanat H, Gökçe F, Şahin E. Self-organized flocking in mobile robot swarms. *Swarm Intelligence* 2008; **2**:97–120.
11. Olfati-saber R. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. on Automatic Control* 2006; **51**:401–420.
12. Fatès N. Solving the decentralised gathering problem with a reaction-diffusion-chemotaxis scheme. *Swarm Intelligence* 2010; **4**:91–115.
13. Yuan X, Su A, Yuan Y, Nie H, Wang L. Non-convex dynamic dispatch of generators with prohibited operating zones. *Optimal Control Applications and Methods* 2009; **30**(1):103–120.
14. Gazi V, Fidan B. Coordination and control of multi-agent dynamic systems: Models and approaches. *Swarm Robotics*, *Lecture Notes in Computer Science*, vol. 4433, 2006; 71–102.
15. Martinez S, Cortes J, Bullo F. Motion coordination with distributed information. *IEEE Control Systems Magazine* 2007; **27**(4):75–88.
16. Gazi V, Passino KM. Stability analysis of swarms. *IEEE Trans. on Automatic Control* Apr 2003; **48**(4):692–697.
17. Fernández Martínez J, García Gonzalo E. The pso family: deduction, stochastic analysis and comparison. *Swarm Intelligence* 2009; **3**:245–273.
18. Allain P, Courty N, Corpetti T. Continuous control of lagrangian data. *Mechanical Engineering and Technology*, *Advances in Intelligent and Soft Computing*, vol. 125, Zhang T (ed.). Springer Berlin Heidelberg, 2012; 307–310.
19. Eberhart R, Kennedy J. A new optimizer using particle swarm theory. *Micro Machine and Human Science, 1995. MHS '95., Nagoya, Proceedings of the Sixth International Symposium on*, 1995; 39 –43.
20. Poli R, Kennedy J, Blackwell T. Particle swarm optimization. *Swarm Intelligence* 2007; **1**:33–57.
21. Lions J. *Optimal Control of Systems Governed by Partial Differential Equations*. Springer-Verlag Berlin, 1971.
22. Pontryagin LS, Boltyanskii VG, Gamkrelidze RV, Mishchenko E. *The mathematical theory of optimal processes (International series of monographs in pure and applied mathematics)*. Interscience Publishers New-York, 1962.
23. Le Dimet FX, Talagrand O. Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. *Tellus A* 1986; **38A**(2):97–110.
24. Talagrand O, Courtier P. Variational assimilation of meteorological observations with the adjoint vorticity equation. i: Theory. *Quarterly Journal of the Royal Meteorological Society* 1987; **113**(478):1311–1328.
25. Thomas C, Corpetti T, Mémin É. Data assimilation for convective-cell tracking on meteorological image sequences. *IEEE T. Geoscience and Remote Sensing* 2010; **48**(8):3162–3177.
26. Allain P, Courty N, Corpetti T. Crowd flow characterization with optimal control theory. *Proceedings of the 9th Asian conference on Computer Vision - Volume Part II*, ACCV'09, Springer-Verlag: Berlin, Heidelberg, 2010; 279–290.
27. Nodet M. Variational assimilation of Lagrangian data in oceanography. *Inverse Problems* 2006; **22**(1):245–263.
28. Diestel R. *Graph Theory*, *Graduate Texts in Mathematics*, vol. 173. 4th edn., Springer, 2010.
29. Wang P. Navigation strategies for multiple autonomous mobile robots moving in formation. *Intelligent Robots and Systems '89, Tsukuba, Japan. The Autonomous Mobile Robots and Its Applications. IROS '89. Proceedings., IEEE/RSJ International Workshop on*, 1989; 486 –493.

30. Ge S, Cui Y. Dynamic motion planning for mobile robots using potential field method. *Autonomous Robots* 2002; **13**:207–222.
31. Niwa HS. Newtonian dynamical approach to fish schooling. *Journal of Theoretical Biology* 1996; **181**(1):47 – 63.
32. Zhang F. Geometric cooperative control of particle formations. *IEEE Trans. on Automatic Control* 2010; **55**(3):800–803.
33. Hamann H, Wörn H. A framework of space–time continuous models for algorithm design in swarm robotics. *Swarm Intelligence* 2008; **2**:209–239.

Table III. Control parameters for the three experiments.

Experiment	$\mathbf{X}$	$\mathbb{M}$						$\mathcal{C}$		$\mathbf{Y}$
	N	$\boldsymbol{\alpha}$	$\mathbf{W}$	$\mathbf{k}$	a	b	$Q_{\mathbf{u}}$	M	o	R^{-1}
KSNA	46	$\vec{0}$	-	$0.\vec{5}$	10	1	0.01	32×32	7	$100N$
Aggregation	64	$0.\vec{5}$	East	$0.\vec{5}$	2	4	0.01	32×32	4	$k_R = \sqrt{10}$
Dipole	64	$0.\vec{5}$	East	$0.\vec{5}$	1	4	0.01	32×32	2	10