Estimation and Control of UAV Swarms for Distributed Monitoring Tasks

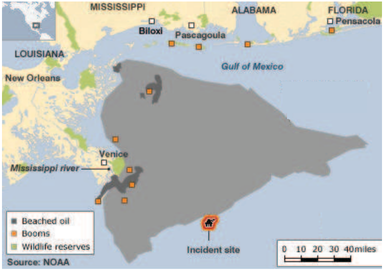
F. Morbidi [⋆], **R.A. Freeman** [◇], **K.M. Lynch** [†]

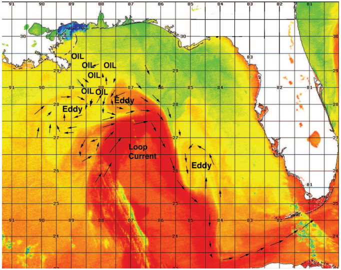
[⋆] Department of Computer Science and Engineering
University of Texas at Arlington, USA

[◇†] Department of Electrical and Mechanical Engineering
Northwestern University, USA

A motivating example

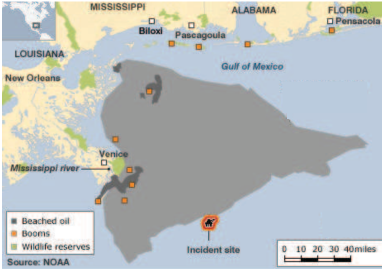
- Lessons learnt from the **Gulf of Mexico blowout** (April 10, 2010):

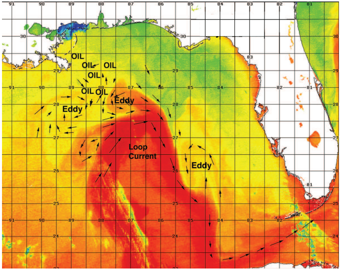




A motivating example

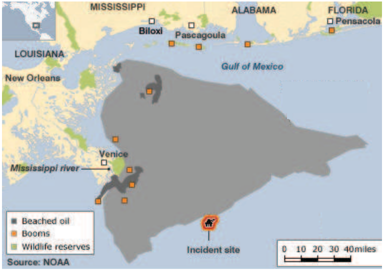
- Lessons learnt from the **Gulf of Mexico blowout** (April 10, 2010):
  1. Difficult to predict the motion of an oil spill. The *direction of sea currents*, *wind intensity*, *evaporation rate*, *oil concentration* **are not precisely known**

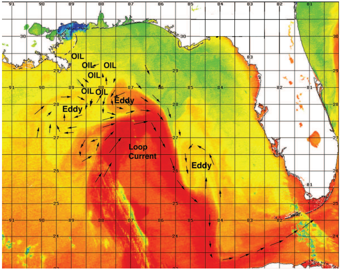




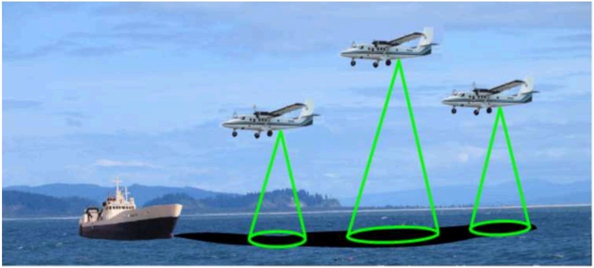
A motivating example

- Lessons learnt from the **Gulf of Mexico blowout** (April 10, 2010):
  1. Difficult to predict the motion of an oil spill. The *direction of sea currents*, *wind intensity*, *evaporation rate*, *oil concentration* **are not precisely known**
  2. It is important to forecast **when** and **where** an oil spill will wash ashore (huge impact on nature reserves, fisheries, tourism, etc.)



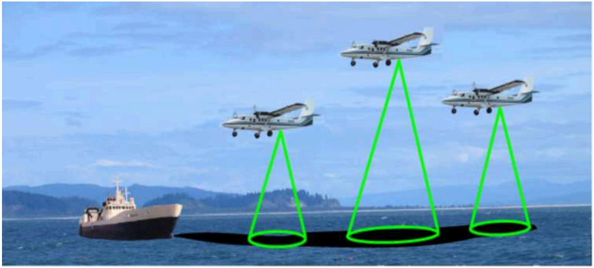


Possible solution



Use a swarm of *unmanned aerial vehicles* (UAVs) for monitoring the oil spill

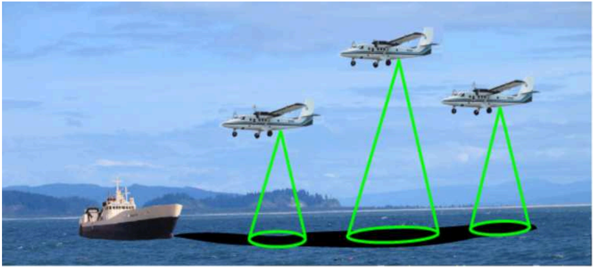
Possible solution



Use a swarm of *unmanned aerial vehicles* (UAVs) for monitoring the oil spill

- Previous approaches → *detection* and *tracking* of the **boundary** of the target region [Casbeer *et al.*, IJSC06], [Susca *et al.*, TCST08], [Smith *et al.*, IJRR10]

Possible solution

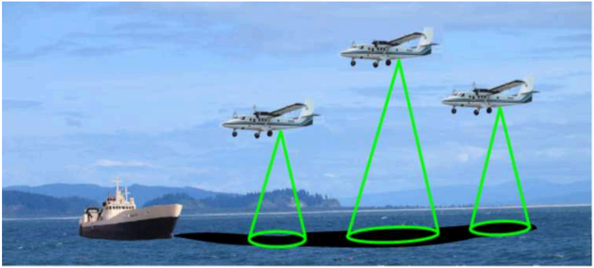


Use a swarm of *unmanned aerial vehicles* (UAVs) for monitoring the oil spill

- Previous approaches → *detection* and *tracking* of the **boundary** of the target region [Casbeer *et al.*, IJSC06], [Susca *et al.*, TCST08], [Smith *et al.*, IJRR10]

  1. The boundary of the target region may be **faint** or **fuzzy** in real settings and thus hard to detect and track

Possible solution

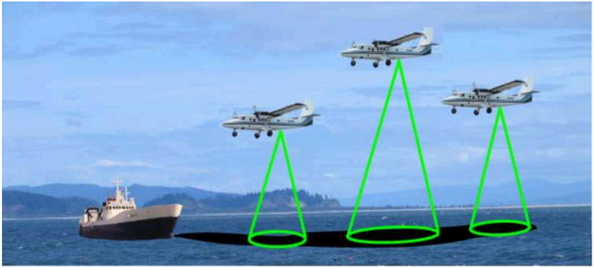


Use a swarm of *unmanned aerial vehicles* (UAVs) for monitoring the oil spill

- Previous approaches → *detection* and *tracking* of the **boundary** of the target region [Casbeer *et al.*, IJSC06], [Susca *et al.*, TCST08], [Smith *et al.*, IJRR10]

  1. The boundary of the target region may be **faint** or **fuzzy** in real settings and thus hard to detect and track
  2. The events occurring at the **center of the target region** are ignored

Possible solution



Use a swarm of *unmanned aerial vehicles* (UAVs) for monitoring the oil spill

- Previous approaches → *detection* and *tracking* of the **boundary** of the target region [Casbeer *et al.*, IJSC06], [Susca *et al.*, TCST08], [Smith *et al.*, IJRR10]

  1. The boundary of the target region may be **faint** or **fuzzy** in real settings and thus hard to detect and track
  2. The events occurring at the **center of the target region** are ignored
  3. The agents are **fully actuated**

Key features of the proposed approach

- The UAVs are **nonholonomic vehicles**

Key features of the proposed approach

- The UAVs are **nonholonomic vehicles**
- The UAVs have **limited sensing capabilities** (i.e., they sense only a *portion* of the environment)

Key features of the proposed approach

- The UAVs are **nonholonomic vehicles**
- The UAVs have **limited sensing capabilities** (i.e., they sense only a *portion* of the environment)
- The target region is described by an ensemble of **particles**

Key features of the proposed approach

- The UAVs are **nonholonomic vehicles**
- The UAVs have **limited sensing capabilities** (i.e., they sense only a *portion* of the environment)
- The target region is described by an ensemble of **particles**
- The "*shape*" of the swarm and the ensemble of particles is described using **geometric moments** [Belta *et al.*, TRO04]

Key features of the proposed approach

- The UAVs are **nonholonomic vehicles**
- The UAVs have **limited sensing capabilities** (i.e., they sense only a *portion* of the environment)
- The target region is described by an ensemble of **particles**
- The "*shape*" of the swarm and the ensemble of particles is described using **geometric moments** [Belta *et al.*, TRO04]

Moments of the swarm $\longleftrightarrow$ **MATCH** $\longleftrightarrow$ Moments of the particles

Key features of the proposed approach

- The UAVs are **nonholonomic vehicles**
- The UAVs have **limited sensing capabilities** (i.e., they sense only a *portion* of the environment)
- The target region is described by an ensemble of **particles**
- The "*shape*" of the swarm and the ensemble of particles is described using **geometric moments** [Belta *et al.*, TRO04]

Moments of the swarm $\longleftrightarrow$ **MATCH** $\longleftrightarrow$ Moments of the particles

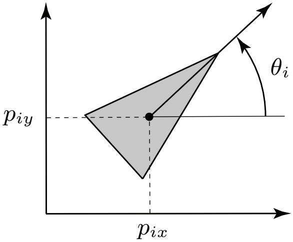
- **Fully distributed** algorithm

Dynamic model of the UAVs

n UAVs flying at *fixed altitude*:

$$\begin{cases} \dot{p}_{ix} = v_i \cos\theta_i \\ \dot{p}_{iy} = v_i \sin\theta_i, \quad i \in \{1, \ldots, n\} \\ \dot{\theta}_i = \omega_i \end{cases}$$

- $\mathbf{p}_i = [p_{ix},\, p_{iy}]^T$: **position** of agent i in the plane of motion
- $\theta_i \in [-\pi,\, \pi)$: **heading** of agent i
- $[v_i,\, \omega_i]^T$, $v_i > 0$: **forward** and **angular velocity** of agent i



Control design: preliminaries

- Let

$$\mathbf{p} = [\mathbf{p}_1^T, \ldots, \mathbf{p}_n^T]^T \in \mathbb{R}^{2n}$$

Control design: preliminaries

- Let

$$\mathbf{p} = [\mathbf{p}_1^T, \ldots, \mathbf{p}_n^T]^T \in \mathbb{R}^{2n}$$

- The *configuration* of the agents is described by the *swarm moment function*:

$$\mathbf{f}(\mathbf{p}) = \frac{1}{n} \sum_{i=1}^{n} \phi(\mathbf{p}_i)$$

Control design: preliminaries

- Let

$$\mathbf{p} = [\mathbf{p}_1^T, \ldots, \mathbf{p}_n^T]^T \in \mathbb{R}^{2n}$$

- The *configuration* of the agents is described by the *swarm moment function*:

$$\mathbf{f}(\mathbf{p}) = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{\phi}(\mathbf{p}_i)$$

- The *moment-generating function* $\boldsymbol{\phi} : \mathbb{R}^2 \to \mathbb{R}^\ell$ is defined as:

$$\boldsymbol{\phi}(\mathbf{p}_i) \triangleq [\, p_{ix},\;\; p_{iy},\;\; p_{ix}^2,\;\; p_{iy}^2,\;\; p_{ix}p_{iy},\;\; p_{ix}^3,\;\; p_{iy}^3,\;\; p_{ix}^2 p_{iy},\; \ldots \,]^T$$

Control design: preliminaries

- Let

$$\mathbf{p} = [\mathbf{p}_1^T, \ldots, \mathbf{p}_n^T]^T \in \mathbb{R}^{2n}$$

- The *configuration* of the agents is described by the *swarm moment function*:

$$\mathbf{f}(\mathbf{p}) = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\phi}(\mathbf{p}_i)$$

- The *moment-generating function* $\boldsymbol{\phi} : \mathbb{R}^2 \to \mathbb{R}^\ell$ is defined as:

$$\boldsymbol{\phi}(\mathbf{p}_i) \triangleq [\underbrace{p_{ix},\ p_{iy},\ p_{ix}^2,\ p_{iy}^2,\ p_{ix}p_{iy}}_{\text{1st and 2nd-order moments}},\ p_{ix}^3,\ p_{iy}^3,\ p_{ix}^2 p_{iy},\ \ldots]^T$$

Control design: preliminaries

Goal: Move the agents so that their final arrangement **minimizes the error**

$$\boxed{\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star}}$$

The **goal vector** $\mathbf{f}^{\star}$ defines the **desired shape** of the formation

Control design: preliminaries

Goal: Move the agents so that their final arrangement **minimizes the error**

$$\boxed{\mathbf{f}(\mathbf{p}) - \mathbf{f}^\star}$$

The **goal vector** $\mathbf{f}^\star$ defines the **desired shape** of the formation

---

For the time being, we assume:

- $\mathbf{f}^\star$ *a priori known*
- $\mathbf{f}^\star$ *constant*

We will relax these hypotheses later on . . .

Nonlinear gradient controller

- Consider the **potential function**:

$$\Pi(\mathbf{p}) = (\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star})^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star})$$

where $\mathbf{\Gamma} \in \mathbb{R}^{5 \times 5}$ is an assigned symmetric positive-definite **gain matrix**

Nonlinear gradient controller

- Consider the **potential function**:

$$\Pi(\mathbf{p}) = (\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star})^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star})$$

where $\mathbf{\Gamma} \in \mathbb{R}^{5\times 5}$ is an assigned symmetric positive-definite **gain matrix**

- Define the vector:

$$\mathbf{g}_i(t) \triangleq -\nabla_{\mathbf{p}_i} \Pi(\mathbf{p}(t)) = -(\mathcal{J}\phi(\mathbf{p}_i(t)))^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}(t)) - \mathbf{f}^{\star})$$

where $\mathcal{J}\phi(\cdot) \in \mathbb{R}^{5\times 2}$ is the **Jacobian matrix** of $\phi(\cdot)$

Nonlinear gradient controller

- Consider the **potential function**:

$$\Pi(\mathbf{p}) = (\mathbf{f}(\mathbf{p}) - \mathbf{f}^\star)^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}) - \mathbf{f}^\star)$$

  where $\mathbf{\Gamma} \in \mathbb{R}^{5\times 5}$ is an assigned symmetric positive-definite **gain matrix**

- Define the vector:

$$\mathbf{g}_i(t) \triangleq -\nabla_{\mathbf{p}_i} \Pi(\mathbf{p}(t)) = -(\mathcal{J}\boldsymbol{\phi}(\mathbf{p}_i(t)))^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}(t)) - \mathbf{f}^\star)$$

  where $\mathcal{J}\boldsymbol{\phi}(\cdot) \in \mathbb{R}^{5\times 2}$ is the **Jacobian matrix** of $\boldsymbol{\phi}(\cdot)$

- Set

$$\alpha_i(t) \triangleq \mathsf{proj}(\arg(\mathbf{g}_i(t)) - \theta_i(t))$$

Nonlinear gradient controller

- Consider the **potential function**:

$$\Pi(\mathbf{p}) = (\mathbf{f}(\mathbf{p}) - \mathbf{f}^\star)^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}) - \mathbf{f}^\star)$$

  where $\mathbf{\Gamma} \in \mathbb{R}^{5\times 5}$ is an assigned symmetric positive-definite **gain matrix**

- Define the vector:

$$\mathbf{g}_i(t) \triangleq -\nabla_{\mathbf{p}_i} \Pi(\mathbf{p}(t)) = -(\mathcal{J}\phi(\mathbf{p}_i(t)))^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}(t)) - \mathbf{f}^\star)$$

  where $\mathcal{J}\phi(\cdot) \in \mathbb{R}^{5\times 2}$ is the **Jacobian matrix** of $\phi(\cdot)$

- Set

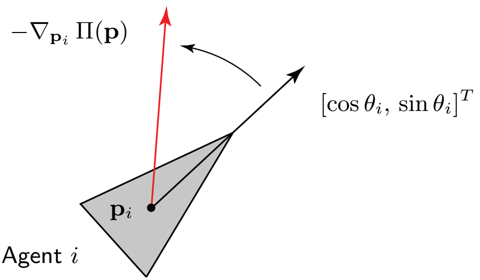
$$\alpha_i(t) \triangleq \text{proj}(\arg(\mathbf{g}_i(t)) - \theta_i(t))$$

- Define the **control input** for agent i as:

$$\boxed{v_i(t) = v, \quad \omega_i(t) = \rho \, \alpha_i(t)}$$

  where v is a **positive constant** and ρ is a **positive gain**

Nonlinear gradient controller: geometric interpretation



- The **angular control** forces the heading direction of agent i to **align with the antigradient** of the potential function $\Pi(\mathbf{p})$

Properties of the controller

$a)$ For any $\epsilon > 0$, there exists a sufficiently large gain ρ such that $\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star}$ is **uniformly ultimately bounded** with an ultimate bound ϵ

Properties of the controller

$a)$ For any $\epsilon > 0$, there exists a sufficiently large gain ρ such that $\mathbf{f}(\mathbf{p}) - \mathbf{f}^\star$ is **uniformly ultimately bounded** with an ultimate bound ϵ

$b)$ For $i, j \in \{1, \ldots, n\}, \;\; i \neq j$, let

$$d_{ij}^{\theta}(t) \triangleq \text{proj}(\theta_i(t) - \theta_j(t))$$
$$d_{ij}^{\omega}(t) \triangleq \omega_i(t) - \omega_j(t)$$

Then, for any $\epsilon_\theta, \, \epsilon_\omega > 0$ there exists a sufficiently large constant $\mu \in \mathbb{R}_{>0}$ satisfying

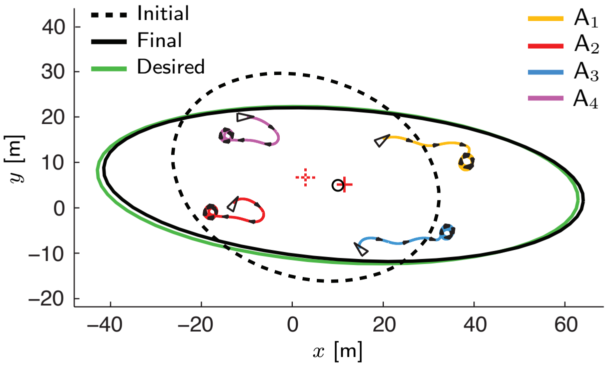
$$\mathbf{\Gamma}[1,1], \, \mathbf{\Gamma}[2,2] \, \geq \, \mu \big| \mathbf{\Gamma}[h,l] \big|, \quad h, l \in \{1, \ldots, 5\}, \;\; (h,l) \neq \{(1,1), \, (2,2)\}$$

such that $|d_{ij}^{\theta}(t)|, \, |d_{ij}^{\omega}(t)|$ are **uniformly ultimately bounded** with ultimate bounds $\epsilon_\theta, \, \epsilon_\omega$

Illustrative example - 1

Trajectory of $n = 4$ agents implementing the gradient controller with:

- $v = 1, \ \rho = 0.5$
- $\mathbf{f}^\star = [10, \ 5, \ 800, \ 100, \ 10]^T$
- $\mathbf{\Gamma} = \text{diag}(1000, 1000, 0.1, 0.1, 0.1)$



Illustrative example - 2

Trajectory of $n = 5$ agents implementing the gradient controller with:

- $v = 1000,\ \rho = 1$
- $\mathbf{f}^{\star} = [10^2,\, 3 \times 10^2,\, 1.7 \times 10^5,\, 0.7 \times 10^5,\, 1.3 \times 10^5]^T$
- $\mathbf{\Gamma} = \text{diag}(10^3, 10^3, 10^{-3}, 10^{-3}, 10^{-3})$

Distributed estimation

- To compute the **angular control**, agent i needs to know $\mathbf{p}$ at each time instant

Distributed estimation

- To compute the **angular control**, agent i needs to know $\mathbf{p}$ at each time instant

  $\Rightarrow$ The control is **not** implementable in a **distributed fashion**

Distributed estimation

- To compute the **angular control**, agent i needs to know $\mathbf{p}$ at each time instant

  $\Rightarrow$ The control is **not** implementable in a **distributed fashion**

- We need suitable *distributed algorithms* to **locally estimate**:

  1. The swarm moment function $\mathbf{f}(\mathbf{p})$

Distributed estimation

- To compute the **angular control**, agent i needs to know $\mathbf{p}$ at each time instant

  $\Rightarrow$ The control is **not** implementable in a **distributed fashion**

- We need suitable *distributed algorithms* to **locally estimate** :

  1. The swarm moment function $\mathbf{f}(\mathbf{p})$
  2. The vector of desired geometric moments using the *environmental data*

Distributed estimation

- To compute the **angular control**, agent i needs to know $\mathbf{p}$ at each time instant

  $\Rightarrow$ The control is **not** implementable in a **distributed fashion**

- We need suitable *distributed algorithms* to **locally estimate**:

  1. The swarm moment function $\mathbf{f}(\mathbf{p})$

  2. The vector of desired geometric moments using the *environmental data*

     We will call it $\mathbf{f}^{\star}_{\text{env}} \longrightarrow$ *environmental goal vector*

Distributed estimation

- Let $\mathbf{q}_k = [q_{kx},\, q_{ky}]^T$ be the *position* of the k-th of N **particles** describing the occurrence of some event of interest in a set $\mathcal{Q} \subset \mathbb{R}^2$ and evolving according to

$$\dot{\mathbf{q}} = \boldsymbol{\Upsilon}(\mathbf{q},\, t), \quad \mathbf{q} = [\mathbf{q}_1^T, \ldots, \mathbf{q}_N^T]^T$$

where $\boldsymbol{\Upsilon} = [\boldsymbol{\Upsilon}_1^T, \ldots, \boldsymbol{\Upsilon}_N^T]^T$ is a *vector field* **unknown** to the agents

Distributed estimation

- Let $\mathbf{q}_k = [q_{kx},\, q_{ky}]^T$ be the *position* of the k-th of N **particles** describing the occurrence of some event of interest in a set $\mathcal{Q} \subset \mathbb{R}^2$ and evolving according to

$$\dot{\mathbf{q}} = \boldsymbol{\Upsilon}(\mathbf{q}, t), \quad \mathbf{q} = [\mathbf{q}_1^T, \ldots, \mathbf{q}_N^T]^T$$

  where $\boldsymbol{\Upsilon} = [\boldsymbol{\Upsilon}_1^T, \ldots, \boldsymbol{\Upsilon}_N^T]^T$ is a *vector field* **unknown** to the agents

- Each agent is equipped with a **limited-footprint sensor**, hence it can measure the x-, y-coordinates of only a **subset** of the N particles

Distributed estimation

- Let $\mathbf{q}_k = [q_{kx},\, q_{ky}]^T$ be the *position* of the k-th of N **particles** describing the occurrence of some event of interest in a set $\mathcal{Q} \subset \mathbb{R}^2$ and evolving according to

$$\dot{\mathbf{q}} = \boldsymbol{\Upsilon}(\mathbf{q},\, t), \quad \mathbf{q} = [\mathbf{q}_1^T, \ldots, \mathbf{q}_N^T]^T$$

where $\boldsymbol{\Upsilon} = [\boldsymbol{\Upsilon}_1^T, \ldots, \boldsymbol{\Upsilon}_N^T]^T$ is a *vector field* **unknown** to the agents

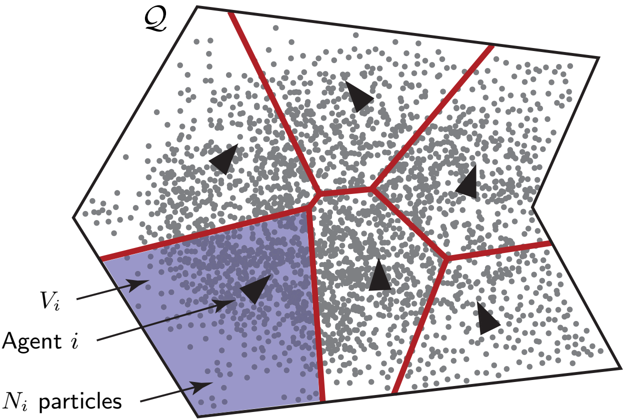
- Each agent is equipped with a **limited-footprint sensor**, hence it can measure the x-, y-coordinates of only a **subset** of the N particles

- **Assumption**: agent i processes **only** the $N_i < N$ particles lying within the **Voronoi cell** V_i that it generates, from which it computes the vector:

$$\boxed{\mathbf{h}_i = \sum_{\mathbf{q}_k :\, \mathbf{q}_k \in V_i} \phi(\mathbf{q}_k)}$$

Distributed estimation

- Voronoi partition of the set $\mathcal{Q}$



Distributed estimation: PI estimators

In order to obtain *local estimates* of $\mathbf{f}(\mathbf{p})$ and of the *environmental goal vector*

$$\mathbf{f}^{\star}_{\mathsf{env}} \triangleq \frac{1}{N}\sum_{k=1}^{N}\phi(\mathbf{q}_k) = \frac{1}{N}\sum_{i=1}^{n}\mathbf{h}_i$$

agent i runs a **proportional-integral (PI) average consensus estimator** [Yang *et al.*, TAC08], [Lynch *et al.*, TRO08]:

$$\dot{\boldsymbol{\xi}}_i = -\gamma\,\boldsymbol{\xi}_i - \sum_{j\neq i}\sigma(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i-\boldsymbol{\xi}_j) + \sum_{j\neq i}\tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\eta}_i-\boldsymbol{\eta}_j) + \gamma\begin{bmatrix}\phi(\mathbf{p}_i)\\ \mathbf{h}_i\\ N_i\end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j\neq i}\tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i-\boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

Distributed estimation: PI estimators

$$\dot{\boldsymbol{\xi}}_i = -\gamma\,\boldsymbol{\xi}_i - \sum_{j\neq i} \sigma(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j) + \sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\eta}_i - \boldsymbol{\eta}_j) + \gamma \begin{bmatrix} \boldsymbol{\phi}(\mathbf{p}_i) \\ \mathbf{h}_i \\ N_i \end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

Distributed estimation: PI estimators

$$\dot{\boldsymbol{\xi}}_i = -\gamma \boldsymbol{\xi}_i - \sum_{j \neq i} \sigma(\mathbf{p}_i, \mathbf{p}_j)(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j) + \sum_{j \neq i} \tau(\mathbf{p}_i, \mathbf{p}_j)(\boldsymbol{\eta}_i - \boldsymbol{\eta}_j) + \gamma \begin{bmatrix} \phi(\mathbf{p}_i) \\ \mathbf{h}_i \\ N_i \end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j \neq i} \tau(\mathbf{p}_i, \mathbf{p}_j)(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

- $[\phi(\mathbf{p}_i)^T, \mathbf{h}_i^T, N_i]^T \in \mathbb{R}^{10} \times \mathbb{Z}_{>0}$: agent i's **input**

Distributed estimation: PI estimators

$$\dot{\boldsymbol{\xi}}_i = -\gamma\,\boldsymbol{\xi}_i - \sum_{j\neq i} \sigma(\mathbf{p}_i, \mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j) + \sum_{j\neq i} \tau(\mathbf{p}_i, \mathbf{p}_j)\,(\boldsymbol{\eta}_i - \boldsymbol{\eta}_j) + \gamma \begin{bmatrix} \boldsymbol{\phi}(\mathbf{p}_i) \\ \mathbf{h}_i \\ N_i \end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j\neq i} \tau(\mathbf{p}_i, \mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

- $[\boldsymbol{\phi}(\mathbf{p}_i)^T, \mathbf{h}_i^T, N_i]^T \in \mathbb{R}^{10} \times \mathbb{Z}_{>0}$: agent i's **input**
- $\boldsymbol{\xi}_i \in \mathbb{R}^{11}$: agent i's **estimate** of the average of all the agents' input

Distributed estimation: PI estimators

$$\dot{\boldsymbol{\xi}}_i = -\gamma\,\boldsymbol{\xi}_i - \sum_{j \neq i} \sigma(\mathbf{p}_i, \mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j) + \sum_{j \neq i} \tau(\mathbf{p}_i, \mathbf{p}_j)\,(\boldsymbol{\eta}_i - \boldsymbol{\eta}_j) + \gamma \begin{bmatrix} \boldsymbol{\phi}(\mathbf{p}_i) \\ \mathbf{h}_i \\ N_i \end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j \neq i} \tau(\mathbf{p}_i, \mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

- $[\boldsymbol{\phi}(\mathbf{p}_i)^T,\, \mathbf{h}_i^T,\, N_i]^T \in \mathbb{R}^{10} \times \mathbb{Z}_{>0}$: agent i's **input**
- $\boldsymbol{\xi}_i \in \mathbb{R}^{11}$: agent i's **estimate** of the average of all the agents' input
- $\boldsymbol{\eta}_i \in \mathbb{R}^{11}$: **internal state** of the PI estimator

Distributed estimation: PI estimators

$$\dot{\boldsymbol{\xi}}_i = -\gamma\,\boldsymbol{\xi}_i - \sum_{j\neq i} \sigma(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j) + \sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\eta}_i - \boldsymbol{\eta}_j) + \gamma \begin{bmatrix} \boldsymbol{\phi}(\mathbf{p}_i) \\ \mathbf{h}_i \\ N_i \end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

- $[\boldsymbol{\phi}(\mathbf{p}_i)^T,\, \mathbf{h}_i^T,\, N_i]^T \in \mathbb{R}^{10} \times \mathbb{Z}_{>0}$: agent i's **input**
- $\boldsymbol{\xi}_i \in \mathbb{R}^{11}$: agent i's **estimate** of the average of all the agents' input
- $\boldsymbol{\eta}_i \in \mathbb{R}^{11}$: **internal state** of the PI estimator
- $\gamma \in \mathbb{R}_{>0}$: global **forgetting factor** governing the rate at which new information replaces the old one in the dynamic averaging process

Distributed estimation: PI estimators

$$\dot{\boldsymbol{\xi}}_i = -\gamma\,\boldsymbol{\xi}_i - \sum_{j\neq i} \sigma(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j) + \sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\eta}_i - \boldsymbol{\eta}_j) + \gamma \begin{bmatrix} \boldsymbol{\phi}(\mathbf{p}_i) \\ \mathbf{h}_i \\ N_i \end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

- $[\boldsymbol{\phi}(\mathbf{p}_i)^T,\, \mathbf{h}_i^T,\, N_i]^T \in \mathbb{R}^{10} \times \mathbb{Z}_{>0}$: agent i's **input**
- $\boldsymbol{\xi}_i \in \mathbb{R}^{11}$: agent i's **estimate** of the average of all the agents' input
- $\boldsymbol{\eta}_i \in \mathbb{R}^{11}$: **internal state** of the PI estimator
- $\gamma \in \mathbb{R}_{>0}$: global **forgetting factor** governing the rate at which new information replaces the old one in the dynamic averaging process
- $\sigma(\mathbf{p}_i,\mathbf{p}_j)$, $\tau(\mathbf{p}_i,\mathbf{p}_j)$: bounded symmetric **gain functions**

Distributed estimation: PI estimators

$$\dot{\boldsymbol{\xi}}_i = -\gamma\,\boldsymbol{\xi}_i - \sum_{j\neq i} \sigma(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j) + \sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\eta}_i - \boldsymbol{\eta}_j) + \gamma \begin{bmatrix} \boldsymbol{\phi}(\mathbf{p}_i) \\ \mathbf{h}_i \\ N_i \end{bmatrix}$$

$$\dot{\boldsymbol{\eta}}_i = -\sum_{j\neq i} \tau(\mathbf{p}_i,\mathbf{p}_j)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_j)$$

$$\boldsymbol{\chi}_i = \boldsymbol{\xi}_i[1:5] - \frac{\boldsymbol{\xi}_i[6:10]}{\boldsymbol{\xi}_i[11]}$$

- $[\boldsymbol{\phi}(\mathbf{p}_i)^T,\, \mathbf{h}_i^T,\, N_i]^T \in \mathbb{R}^{10} \times \mathbb{Z}_{>0}$: agent i's **input**
- $\boldsymbol{\xi}_i \in \mathbb{R}^{11}$: agent i's **estimate** of the average of all the agents' input
- $\boldsymbol{\eta}_i \in \mathbb{R}^{11}$: **internal state** of the PI estimator
- $\gamma \in \mathbb{R}_{>0}$: global **forgetting factor** governing the rate at which new information replaces the old one in the dynamic averaging process
- $\sigma(\mathbf{p}_i,\mathbf{p}_j),\ \tau(\mathbf{p}_i,\mathbf{p}_j)$: bounded symmetric **gain functions**
- $\boldsymbol{\chi}_i \in \mathbb{R}^5$: **output** of the PI estimator $\longrightarrow$ agent i's estimate of $\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star}_{\text{env}}$

Closed-loop stability

Theorem - (Main result)

Suppose that the topology of the network remains **always connected** in forward time. Suppose that $n \geq 3$ is fixed and that the **control input** of agent i is of the form

$$v_i(t) = v, \quad \omega_i(t) = \rho\, \alpha_i(t)$$

with $\alpha_i(t) \triangleq \text{proj}(\arg(\mathbf{g}_i(t)) - \theta_i(t))$ and

$$\mathbf{g}_i(t) = -(\mathcal{J}\boldsymbol{\phi}(\mathbf{p}_i(t)))^T\, \boldsymbol{\Gamma}\, \boldsymbol{\chi}_i(t)$$

Let us also suppose that $\|\boldsymbol{\Upsilon}_k(\mathbf{q}, t)\|$, $\forall\, k \in \{1, \ldots, N\}$, is sufficiently smaller than v. Then, for almost every initial configuration of the agents:

- Each *trajectory* of the swarm system is **bounded in forward time**
- For any $\epsilon > 0$, there exists a sufficiently large gain ρ such that the error $\mathbf{f}(\mathbf{p}) - \mathbf{f}^*_{\text{env}}$ is **uniformly ultimately bounded** with an ultimate bound ϵ

Closed-loop stability

Theorem - (Main result)

Suppose that the topology of the network remains **always connected** in forward time. Suppose that $n \geq 3$ is fixed and that the **control input** of agent i is of the form

$$v_i(t) = v, \quad \omega_i(t) = \rho\, \alpha_i(t)$$

with $\alpha_i(t) \triangleq \mathrm{proj}(\arg(\mathbf{g}_i(t)) - \theta_i(t))$ and

$$\mathbf{g}_i(t) = -(\mathcal{J}\boldsymbol{\phi}(\mathbf{p}_i(t)))^T\, \boldsymbol{\Gamma}\, \boldsymbol{\chi}_i(t)$$

Let us also suppose that $\|\boldsymbol{\Upsilon}_k(\mathbf{q}, t)\|$, $\forall\, k \in \{1, \ldots, N\}$, is sufficiently smaller than v. Then, for almost every initial configuration of the agents:

- Each *trajectory* of the swarm system is **bounded in forward time**
- For any $\epsilon > 0$, there exists a sufficiently large gain ρ such that the error $\mathbf{f}(\mathbf{p}) - \mathbf{f}^*_{\mathrm{env}}$ is **uniformly ultimately bounded** with an ultimate bound ϵ

Proof: It leverages the *small-gain theorem*

Simulation results: control + estimation

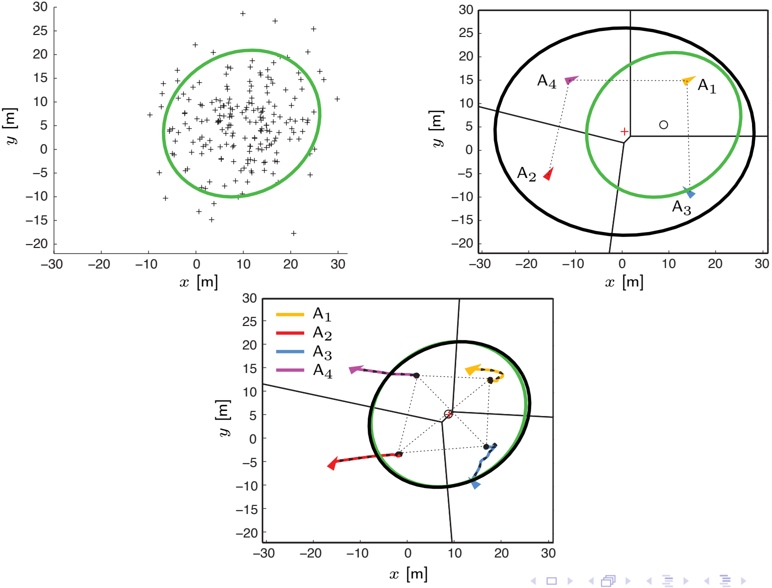
- $n = 4$ agents
- **Gradient controllers**: $v = 1$, $\rho = 3$ and $\mathbf{\Gamma} = \text{diag}(100, 100, 0.1, 0.1, 0.1)$
- **PI estimators**: $\gamma = 7$ and $\sigma(\cdot,\cdot)$, $\tau(\cdot,\cdot)$ are chosen according to an *equal weighting scheme* with a communication radius $R = 27$ m:

$$\begin{cases} \sigma(\mathbf{p}_i, \mathbf{p}_j) = 25 \;\text{ and }\; \tau(\mathbf{p}_i, \mathbf{p}_j) = 0.8, & \text{if } \|\mathbf{p}_i - \mathbf{p}_j\| \leq R \\ \sigma(\mathbf{p}_i, \mathbf{p}_j) = \tau(\mathbf{p}_i, \mathbf{p}_j) = 0, & \text{otherwise} \end{cases}$$

- $N = 200$ particles drawn from a **bivariate normal distribution** $\mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ with:

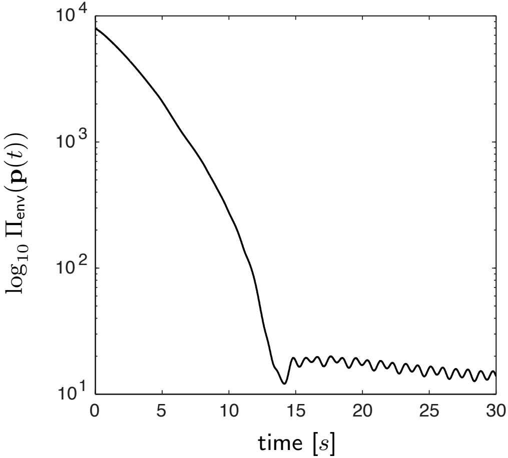
$$\boldsymbol{\mu} = \begin{bmatrix} 10 \\ 5 \end{bmatrix}, \qquad \mathbf{\Sigma} = \begin{bmatrix} 70 & 1 \\ 1 & 70 \end{bmatrix}$$

Simulation results: control + estimation



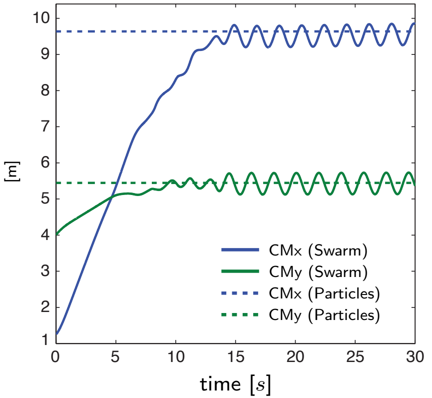
Simulation results: control + estimation

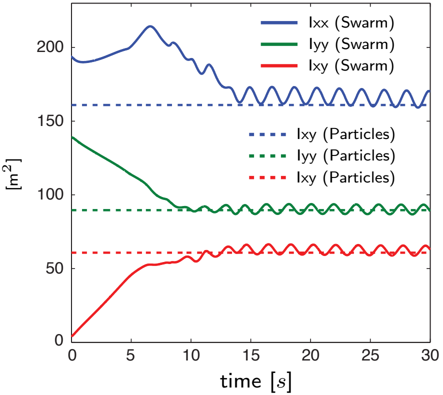
- Time history of $\Pi_{\text{env}}(\mathbf{p}) = (\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star}_{\text{env}})^T \, \mathbf{\Gamma} \, (\mathbf{f}(\mathbf{p}) - \mathbf{f}^{\star}_{\text{env}})$



Simulation results: control + estimation

- $\mathbf{f}^{\star}_{\text{env}}(t)$ (dashed) and $\mathbf{f}(\mathbf{p}(t))$ (solid): *1st-order* (left) and *2nd-order moments* (right)





Conclusions

- New **estimation-and-control strategy** for distributed monitoring tasks

Conclusions

- New **estimation-and-control strategy** for distributed monitoring tasks
- Swarm of UAVs modeled as **constant-speed unicycles**

Conclusions

- New **estimation-and-control strategy** for distributed monitoring tasks
- Swarm of UAVs modeled as **constant-speed unicycles**
- The geometric moments of the swarm are controlled via a **nonlinear gradient descent** to *match* those of an ensemble of particles describing the occurrence of events of interest to be monitored

Conclusions

- New **estimation-and-control strategy** for distributed monitoring tasks
- Swarm of UAVs modeled as **constant-speed unicycles**
- The geometric moments of the swarm are controlled via a **nonlinear gradient descent** to *match* those of an ensemble of particles describing the occurrence of events of interest to be monitored
- **PI average consensus estimators** are used to make the control implementable in a *distributed fashion*

Conclusions

- New **estimation-and-control strategy** for distributed monitoring tasks
- Swarm of UAVs modeled as **constant-speed unicycles**
- The geometric moments of the swarm are controlled via a **nonlinear gradient descent** to *match* those of an ensemble of particles describing the occurrence of events of interest to be monitored
- **PI average consensus estimators** are used to make the control implementable in a *distributed fashion*
- **Closed-loop** stability analysis

Future challenges

- Extension of our strategy to $SE(3)$ and to vehicles with **non-constant** positive forward velocity

Future challenges

- Extension of our strategy to SE(3) and to vehicles with **non-constant** positive forward velocity
- Use **2nd-order central moments** in order to have a *translation-invariant* description of the desired swarm configuration

Future challenges

- Extension of our strategy to $SE(3)$ and to vehicles with **non-constant** positive forward velocity
- Use **2nd-order central moments** in order to have a *translation-invariant* description of the desired swarm configuration
- Test our estimation-and-control algorithm on **real data** (e.g., on *recorded* or *simulated trajectories* of marine oil spills)