Self-triggered coordination of robotic networks for optimal deployment

Jorge Cortés

Mechanical and Aerospace Engineering
University of California, San Diego
http://tintoretto.ucsd.edu/jorge

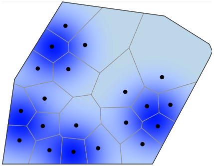
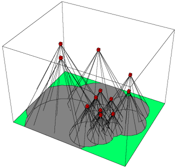
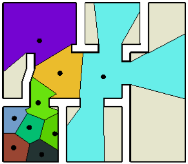
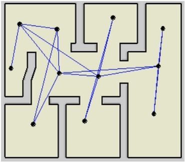
**2011 Santa Barbara Control Workshop:
Decision, Dynamics and Control in Multiagent Systems**

University of California, Santa Barbara
June 24, 2011

Joint work with **Cameron Nowzari**

Research challenges

Adaptive/reliable	robust, predictable behavior
Interactions	who knows what, when, why, how, dynamically changing, no omniscient leader
Uncertainty	imprecise information, unknown environment, events, evolving tasks
Performance	autonomous, efficient, trade-offs

Self-triggered control as a tool for addressing challenges

-Self-triggered- control

Given desired task,

$$\text{information} \xrightarrow{\text{design}} \text{agent plans} \xrightarrow{\text{executions}} \text{performance}$$

What is **necessary information** to achieve desired performance level?

Answer allows for **self-triggered strategies**: what info, how up-to-date, to achieve task within specified performance level

Benefits

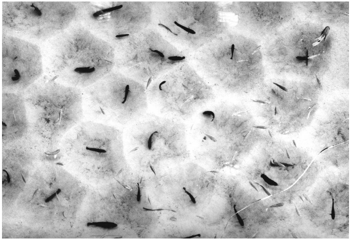
- incorporates **uncertainty** at the design and planning stage
- handles **asynchronous** executions of plans
- **energy savings** in communication/sensing
- **more computation** and decision making
- **less exposure** to detection by adversaries

-optimal deployment- of robotic sensor networks

Objective: optimal task allocation and space partitioning
optimal placement and tuning of sensors

Why?

- servicing
- resource allocation
- environmental monitoring
- data collection
- force protection
- surveillance



Outline

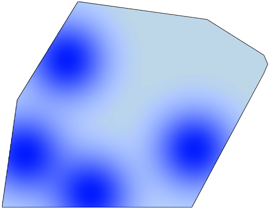
1. motivation
2. deployment
   - aggregate objective optimization
   - Voronoi partition
   - `centroid algorithm`
3. self-triggered deployment
   - guaranteed Voronoi diagrams
   - `self-triggered centroid algorithm`
   - convergence analysis

Expected-value multicenter function

Objective: Given sensors/nodes/robots/sites $(p_1, \ldots, p_n)$ moving in environment S achieve **optimal coverage**

$\phi \colon \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ density

agent performance decreases with distance



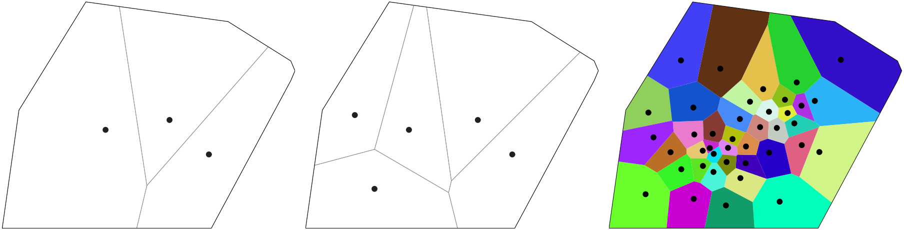
$$\text{minimize} \quad \mathcal{H}(p_1, \ldots, p_n) = E_\phi \left[\min_{i \in \{1,\ldots,n\}} \|q - p_i\|^2 \right]$$

Voronoi partitions

Let $(p_1, \dots, p_n) \in S^n$ denote the positions of n points

The **Voronoi partition** $\mathcal{V}(P) = \{V_1, \dots, V_n\}$ generated by $(p_1, \dots, p_n)$

$$\begin{aligned} V_i &= \{q \in S \mid \ \|q - p_i\| \leq \|q - p_j\|, \ \forall j \neq i\} \\ &= S \cap_j \mathcal{HP}(p_i, p_j) \qquad \text{where } \mathcal{HP}(p_i, p_j) \text{ is half plane } (p_i, p_j) \end{aligned}$$



3 generators　　5 generators　　50 generators

Optimal configurations of $\mathcal{H}$

Alternative expression in terms of Voronoi partition,

$$\mathcal{H}(p_1, \dots, p_n) = \sum_{i=1}^{n} \int_{V_i} \|q - p_i\|_2^2 \phi(q) dq$$

$\mathcal{H}$ as a function of agent positions and partition,

$$\mathcal{H}(p_1, \dots, p_n, W_1, \dots, W_n) = \sum_{i=1}^{n} \int_{W_i} f(\|q - p_i\|_2) \phi(q) dq$$

For fixed positions, **Voronoi partition is optimal**

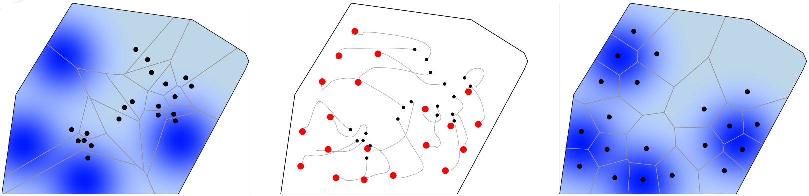
For fixed partition, **centroid configurations are optimal**

centroid algorithm
[Cortes, Martinez, Karatas, Bullo, 04]

At each round, agents synchronously execute:

- transmit position and receive neighbors' positions;
- compute centroid of own cell

Between communication rounds, each robot moves toward center



initial configuration   gradient descent   final configuration

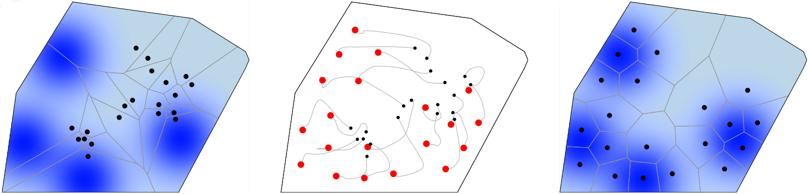
Properties: provably correct, adaptive, distributed over Voronoi graph

geometric-center algorithm
[Cortes, Martinez, Karatas, Bullo, 04]

At each round, agents synchronously execute:

- transmit position and receive neighbors' positions;
- compute notion of geometric center of own cell

Between communication rounds, each robot moves toward center



initial configuration   gradient descent   final configuration

Properties: provably correct, adaptive, distributed over Voronoi graph

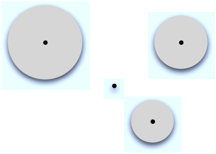
Trading computation for communication/sensing

Balance cost of up-to-date information with limited resources

what can agents do with outdated information about each other?

Agents have **uncertainty regions** on other agents

- how up-to-date information must be to positively contribute to task
- when information must be updated



Each agent i stores $\mathcal{D}^i = ((p_1^i, r_1^i), \dots, (p_n^i, r_n^i))$,

- p_j^i : last known location of agent j
- r_j^i : maximum distance traveled by agent j since last info
- $p_i^i = p_i$ and $r_i^i = 0$

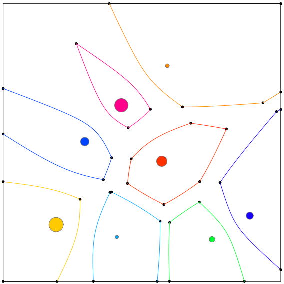
Agents move at max speed $v_{\max}$

Guaranteed Voronoi diagram
[Sember and Evans, 08]

Guaranteed Voronoi diagram $\mathrm{g}\mathcal{V}(D_1, \dots, D_n) = \{\mathrm{g}V_1, \dots, \mathrm{g}V_n\}$ of S generated by $D_1, \dots, D_n \subset S$,

$$\mathrm{g}V_i = \{q \in S \mid \max_{x \in D_i} \|q - x\|_2 \leq \min_{y \in D_j} \|q - y\|_2 \text{ for all } j \neq i\}$$

$\mathrm{g}V_i$ contains points guaranteed to be closer to any point in D_i than to any other point in D_j, $j \neq i$



In general, for $p_i \in D_i$, $\mathrm{g}V_i \subset V_i$

For $D_i = \overline{B}_{r_i}(x_i)$, $\partial \mathrm{g}V_i$ union of hyperbolic arms,

$\{q \in S \mid \|q - x_i\|_2 + r_i = \|q - x_j\|_2 - r_j\}$

Dual guaranteed Voronoi diagram

Dual guaranteed Voronoi diagram $\mathrm{dg}\mathcal{V}(D_1, \dots, D_n) = \{\mathrm{dg}V_1, \dots, \mathrm{dg}V_n\}$ of S generated by $D_1, \dots, D_n \subset S$,

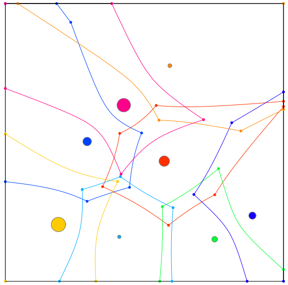
$$\mathrm{dg}V_i = \{q \in S \mid \min_{x \in D_i} \|q - x\|_2 \leq \max_{y \in D_j} \|q - y\|_2 \text{ for all } j \neq i\}$$

Points outside $\mathrm{dg}V_i$ are guaranteed to be closer to any point of D_j than to any point of D_i

In general, for $p_i \in D_i$, $V_i \subset \mathrm{dg}V_i$

For $D_i = \overline{B}_{r_i}(x_i)$, $\partial \mathrm{dg}V_i$ is union of hyperbolic arms,

$\{q \in S \mid \|q - x_i\|_2 - r_i = \|q - x_j\|_2 + r_j\}$



When is motion good?

With outdated info, agent i cannot calculate $\mathsf{cntr}(V_i)$

Proposition

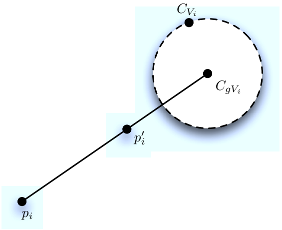
Let $L \subset V \subset U$. Then, for any density function ϕ,

$$\|\,\mathsf{cntr}(V) - \mathsf{cntr}(L)\|_2 \leq \mathit{bound}(L, U) = 2\,\mathsf{cr}(U)\Big(1 - \frac{\mathsf{mass}(L)}{\mathsf{mass}(U)}\Big)$$

Agent i moves from p_i to p_i' making sure that

$$\begin{aligned}\|p_i' - C_{\mathrm{g}V_i}\|_2 &\geq \mathtt{bound}_i = \mathtt{bound}(\mathrm{g}V_i, \mathrm{dg}V_i) \\ &\geq \|C_{V_i} - C_{\mathrm{g}V_i}\|_2\end{aligned}$$

As time elapses without new info, $\mathtt{bound}$ grows



one-step-ahead update decision policy

Agent $i \in \{1, \ldots, n\}$ performs:

set $D = \mathcal{D}^i$
compute $L = \mathrm{g}V_i(D)$ and $U = \mathrm{dg}V_i(D)$ `--guaranteed cells`
compute $q = C_L$ and $r = \mathsf{bound}(L, U)$
if $r \geq \max\{\|q - p_i\|_2, \epsilon\}$ **then**
    update $\mathcal{D}^i$ `--get fresh info`
else
    set $\mathcal{D}^i_j = (p^i_j, r^i_j + v_{\max}\Delta t)$, $j \neq i$ `--increase uncertainty`
    set $\mathcal{D}^i_i = (\mathsf{tbb}(p_i, v_{\max}, q, r), 0)$ `--new position`
end if

Agents can also autonomously schedule updates in the future via `multiple-steps-ahead update decision policy`

one-step-ahead update decision policy

Agent $i \in \{1, \dots, n\}$ performs:

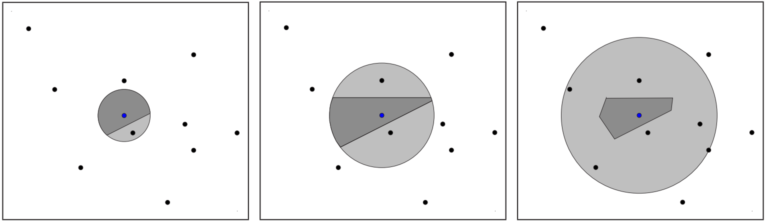
set $D = \mathcal{D}^i$
compute $L = \mathrm{g}V_i(D)$ and $U = \mathrm{dg}V_i(D)$ --guaranteed cells
compute $q = C_L$ and $r = \mathsf{bound}(L, U)$
if $r \geq \max\{\|q - p_i\|_2, \epsilon\}$ **then**
  update $\mathcal{D}^i$ --get fresh info
else
  set $\mathcal{D}^i_j = (p^i_j, r^i_j + v_{\max}\Delta t)$, $j \neq i$ --increase uncertainty
  set $\mathcal{D}^i_i = (\mathsf{tbb}(p_i, v_{\max}, q, r), 0)$ --new position
end if

Agents can also autonomously schedule updates in the future via `multiple-steps-ahead update decision policy`

How to update the agent memory?

Brute-force update mechanism: agent acquires up-to-date information about the location of everybody else in the network – costly, not scalable

Alternative update mechanism: keep track only of a subset of agents, $\mathcal{A}^i$, and update via `Voronoi cell computation`



Voronoi cell computation

[Cortes, Martinez, Karatas, Bullo, 04]

At $\ell \in \mathbb{Z}_{\geq 0}$, agent $i \in \{1, \dots, n\}$ performs:

1: initialize $R_i = \min_{k \in \{1,\dots,n\} \setminus \{i\}} \|p_i - p_k^i\|_2 + v_{\max} \tau_k^i$
2: detect p_j within R_i, set $W(p_i, R_i) = \overline{B}_{R_i}(p_i) \cap \left(\cap_{j : \|p_i - p_j\| \leq R_i} H_{p_i p_j} \right)$
3: **while** $R_i < 2 \max_{q \in W(p_i, R_i)} \|p_i - q\|$ **do**
4: set $R_i := 2R_i$
5: detect all p_j radius R_i, recompute $W(p_i, R_i)$
6: **end while**
7: set $V_i = W(p_i, R_i)$, $\mathcal{A}^i = \mathcal{N}_i \cup \{i\}$ and $\mathcal{D}_j^i = (p_j, 0)$ for $j \in \mathcal{N}_i$

Lemma

Info on agents in $\mathcal{A}^i$ is enough to compute, at each timestep between updates,

- *exact guaranteed Voronoi cell*
- *upper bound of dual guaranteed Voronoi cell*

Proof is nice consequence of geometric properties of guaranteed diagrams

self-triggered centroid algorithm

`self-triggered centroid algorithm` combines

- motion law: `motion control law`
- update policy: `one-step-ahead update decision policy` or `multiple-steps-ahead update decision policy`
- up-to-date information: `Voronoi cell computation`

Proposition

$\mathcal{H}$ is monotonically nonincreasing

Essentially, because algorithm guarantees each agent gets closer to centroid

What about **asymptotic behavior?**

Convergence guarantees

`self-triggered centroid algorithm` is not amenable to standard discrete-time stability analysis because

$$f_{\mathsf{stca}} = f_{\mathsf{motion}} \circ f_{\mathsf{inf}} \quad \text{is discontinuous}$$

f_{inf} "decide/acquire-up-to-date-information"
f_{motion} "move-and-update-uncertainty"

Our strategy

1. construct **set-valued map** T whose trajectories include the ones of f_{stca}
2. make sure T has the **right "continuity"** properties (T closed)
3. analyze with **set-valued** discrete-time stability **analysis**

Embedding the trajectories of f_{stca}

Motion & uncertainty update: continuous $\mathcal{M}\colon (S \times \mathbb{R}_{\geq 0})^{n^2} \to (S \times \mathbb{R}_{\geq 0})^{n^2}$

$$\mathcal{M}_i(\mathcal{D}) = \big((p_1^i, r_1^i + v_{\max}\Delta t_{|\,\mathsf{diam}(S)}, \dots,$$
$$(\mathsf{tbb}(p_i^i, v_{\max}, C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i)), \mathsf{bound}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))), 0), \dots, (p_n^i, r_n^i + v_{\max}\Delta t_{|\,\mathsf{diam}(S)}\big)$$

Embedding the trajectories of f_{stca}

Motion & uncertainty update: continuous $\mathcal{M}\colon (S \times \mathbb{R}_{\geq 0})^{n^2} \to (S \times \mathbb{R}_{\geq 0})^{n^2}$

$$\mathcal{M}_i(\mathcal{D}) = \big((p_1^i, r_1^i + v_{\max}\Delta t_{|\,\mathsf{diam}(S)}, \dots,$$
$$(\mathsf{tbb}(p_i^i, v_{\max}, C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i)), \mathsf{bound}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))), 0), \dots, (p_n^i, r_n^i + v_{\max}\Delta t_{|\,\mathsf{diam}(S)}\big)$$

Acquire up-to-date information: closed $\mathcal{U} : (S \times \mathbb{R}_{\geq 0})^{n^2} \rightrightarrows (S \times \mathbb{R}_{\geq 0})^{n^2}$
$\mathcal{U}(\mathcal{D})$ is Cartesian product whose ith component can be

$\mathcal{D}^i$	i does not get up-to-date information
$((p'_1, r'_1), \dots, (p'_n, r'_n))$	i gets up-to-date information

with $(p'_j, r'_j) = (p_j^j, 0)$ for $j \in \{i\} \cup \mathcal{N}_i$ and $(p'_j, r'_j) = (p_j^i, r_j^i)$ otherwise

Embedding the trajectories of f_{stca}

Motion & uncertainty update: continuous $\mathcal{M}\colon (S \times \mathbb{R}_{\geq 0})^{n^2} \to (S \times \mathbb{R}_{\geq 0})^{n^2}$

$$\mathcal{M}_i(\mathcal{D}) = \big((p_1^i, r_1^i + v_{\max}\Delta t_{|\,\mathsf{diam}(S)}, \dots,$$
$$(\mathsf{tbb}(p_i^i, v_{\max}, C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i)), \mathsf{bound}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))), 0), \dots, (p_n^i, r_n^i + v_{\max}\Delta t_{|\,\mathsf{diam}(S)}\big)$$

Acquire up-to-date information: closed $\mathcal{U} : (S \times \mathbb{R}_{\geq 0})^{n^2} \rightrightarrows (S \times \mathbb{R}_{\geq 0})^{n^2}$
$\mathcal{U}(\mathcal{D})$ is Cartesian product whose ith component can be

$$\begin{array}{ll} \mathcal{D}^i & i \text{ does not get up-to-date information} \\ ((p'_1, r'_1), \dots, (p'_n, r'_n)) & i \text{ gets up-to-date information} \end{array}$$

with $(p'_j, r'_j) = (p_j^j, 0)$ for $j \in \{i\} \cup \mathcal{N}_i$ and $(p'_j, r'_j) = (p_j^i, r_j^i)$ otherwise

Set-valued map $T = \mathcal{U} \circ \mathcal{M} : (S \times \mathbb{R}_{\geq 0})^{n^2} \rightrightarrows (S \times \mathbb{R}_{\geq 0})^{n^2}$ has properties

1. if $\gamma = \{\mathcal{D}(t_\ell)\}$ evolution of f_{stca}, $\gamma' = \{\mathcal{D}'(t_\ell) = f_{\mathsf{inf}}(\mathcal{D}(t_\ell))\}$ evolution of T
2. T is closed

Establishing correctness

Proposition

For $\epsilon \in [0, \mathrm{diam}(S))$, network evolving under **`self-triggered centroid algorithm`** *from any initial configuration in S^n converges the set of centroidal Voronoi configurations, while monotonically optimizing $\mathcal{H}$*

Establishing correctness

Proposition

For $\epsilon \in [0, \mathsf{diam}(S))$, *network evolving under* ***self-triggered centroid algorithm*** *from any initial configuration in* S^n *converges the set of centroidal Voronoi configurations, while monotonically optimizing* $\mathcal{H}$

Proof sketch: for $\gamma' = f_{\mathsf{inf}}(\gamma)$,

- use weak positively T-invariance of $\Omega(\gamma')$ to show

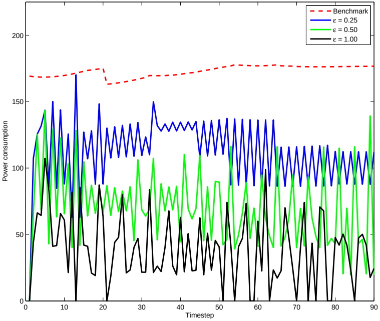
$$\Omega(\gamma') \subseteq \{\mathcal{D} \in (S \times \mathbb{R}_{\geq 0})^{n^2} \mid \|p_i^i - C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\|_2 \leq \mathsf{bound}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\}$$

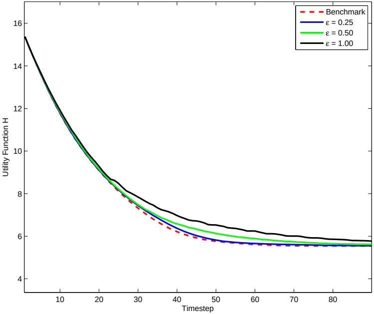
- $\mathsf{bound}_i < \max\{\|p_i^i - C_{\mathrm{g}V_i}\|_2, \epsilon\}$ on γ' and continuity imply on $\Omega(\gamma')$

$$\mathsf{bound}(\pi_{\mathcal{A}^i}(\mathcal{D}^i)) \leq \max\{\|p_i^i - C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\|_2, \epsilon\}$$

- combine facts to show $\Omega(\gamma') \subseteq \{\mathcal{D} \in (S \times \mathbb{R}_{\geq 0})^{n^2} \mid p_i^i = C_{V_i}\}$

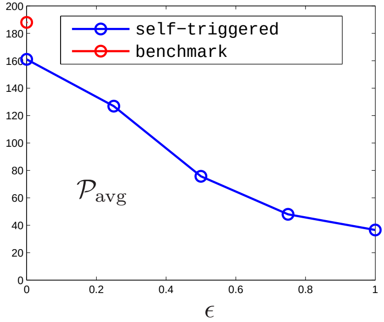
Communication cost and performance
density is sum of two Gaussians

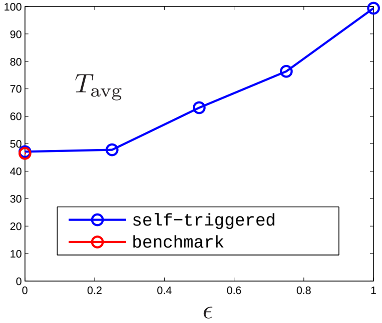




Average communication cost and time to convergence

density is sum of two Gaussians, 20 executions, random initial conditions





Conclusions

Self-triggered optimal **deployment** of robotic sensor networks

- `self-triggered centroid algorithm`
- correct, adaptive, distributed, uncertainty
- same convergence guarantees as synchronous algorithm with perfect information at all times
- extensions to asynchronous executions and dynamically changing $v_{\max}$

Towards **self-triggered coordination**

- applications to other collective tasks: servicing, routing, detection, queuing
- analytical characterization of trade-offs
- broadly applicable mathematical tools