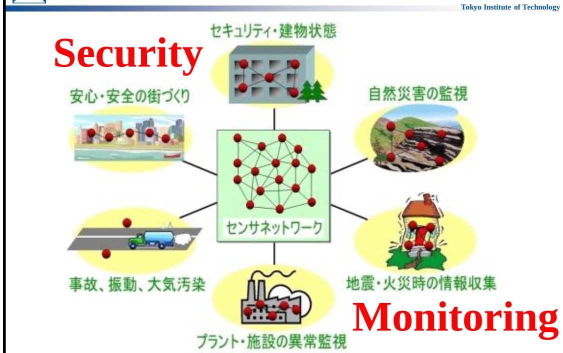
Locational Optimization Problems on Sensor Networks

FL07-30-2
Tatsuya Miyano

Outline

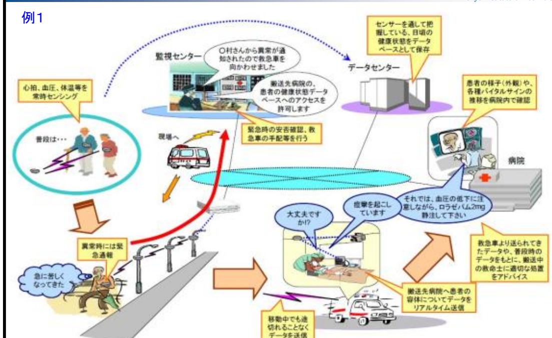
1. Introduction
2. Asynchronous Coverage Control
3. Optimal Assignment for Routing
4. Conclusions and Future Works

Introduction to Sensor Networks



Introduction to Sensor Networks

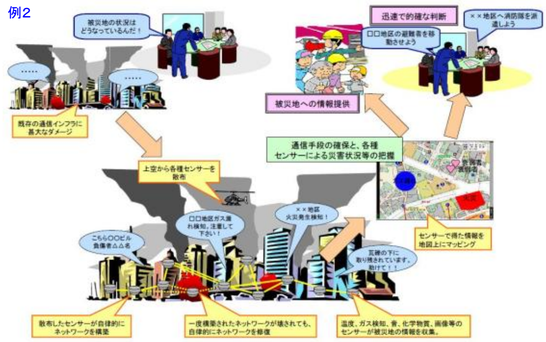
例1



総務省　ユビキタスセンサネットワーク技術に関する調査研究会報告書（平成16年7月）

Introduction to Sensor Networks

例2



総務省　ユビキタスセンサネットワーク技術に関する調査研究会報告書（平成16年7月）

Introduction to Sensor Networks

Researches on Sensor Networks

- Ad Hoc Networks
  - Media Access Control
  - Routing Algorithm *et al.*
- Locational Optimization ← We focus on this topic.
  - Topology Control
  - Coverage Control *et al.*
- Network-Based Control *et al.*

In particular

- Asynchronous Coverage Control
- Optimal Assignment for Routing

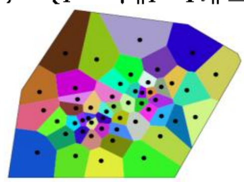
Outline

Coverage Control

Voronoi Region

Given $S \subset \mathbb{R}^2$ and a set $\mathcal{P} = \{p_1, p_2, \ldots, p_n\} \subset S$ of n distinct points, the Voronoi region of S generated by $\mathcal{P}$ is the collection of sets $\{V_1(\mathcal{P}), V_2(\mathcal{P}) \ldots, V_n(\mathcal{P})\}$ defined by

$$V_i(\mathcal{P}) = \{q \in S \mid \|q - p_i\| \le \|q - p_j\|, \ {}^\forall p_j \in \mathcal{P}\}.$$



S. Martinez, J. Cortes and F. Bullo, "Motion Coordination with Distributed Information," *IEEE Control Systems Magazine*, Vol. 27, No. 4, pp. 75-88, 2007.

Coverage Control

Distortion Problem $\quad \min_{p_1, p_2, \ldots, p_n} H(\mathcal{P})$

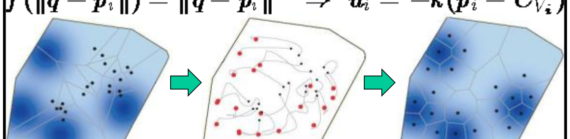
$f(\|q - p_i\|)$: Performance Function $\quad H(\mathcal{P})$: Distortion Function

$\phi(q)$: Density Function $\quad C_{V_i}$: Centroid of i-th Voronoi Region

$$H(\mathcal{P}) = \sum_{i=1}^{n} \int_{V_i(\mathcal{P})} f(\|q - p_i\|)\phi(q)dq \ \Rightarrow \ \dot{p}_i = u_i, \quad u_i = -\frac{\partial H}{\partial p_i}$$

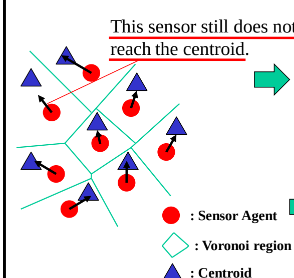
e.g.

$$f(\|q - p_i\|) = \|q - p_i\|^2 \ \Rightarrow \ u_i = -k(p_i - C_{V_i})$$



S. Martinez, J. Cortes and F. Bullo, "Motion Coordination with Distributed Information," *IEEE Control Systems Magazine*, Vol. 27, No. 4, pp. 75-88, 2007.

Asynchronous Coverage Control



What could we do ?

- We could wait for this sensor until it reaches the centroid.
- We could update Voronoi regions.

But

- Time to wait is cost, isn't it ?
- When should we update Voronoi regions ?

We propose an algorithm that considers time cost and update timing.

Time-Space Voronoi Region

Time-Space Voronoi Region

$$V_i^{ts}(\mathcal{P}, \mathcal{T}) := \left\{ \begin{bmatrix} q \\ t \end{bmatrix} \in S \times \mathbb{R}_+ \ \middle| \ \left\| \begin{bmatrix} q \\ ct \end{bmatrix} - \begin{bmatrix} p_i(t + \tau_i(t)) \\ c(t + \tau_i(t)) \end{bmatrix} \right\| \le \left\| \begin{bmatrix} q \\ ct \end{bmatrix} - \begin{bmatrix} p_j(t + \tau_j(t)) \\ c(t + \tau_j(t)) \end{bmatrix} \right\|, \begin{bmatrix} {}^\forall p_j(t + \tau_j(t)) \\ {}^\forall t \end{bmatrix} \in \mathcal{P} \times \mathbb{R}_+ \right\}$$

$\mathcal{T} = \{\tau_1(t), \tau_2(t), \ldots, \tau_n(t)\} \quad \mathbb{R}_+ \ni c$: Parameter

$\tau_i(t)$: Sensing Interval of i-th Sensor Agent at time t

Asynchronous Distortion Problem

Asynchronous Distortion Problem

$$\min_{\substack{p_1(t), p_2(t), \ldots, p_n(t) \\ \tau_1(t), \tau_2(t), \ldots, \tau_n(t)}} H_{ts}(\mathcal{P}, \mathcal{T})$$

$f_{ts}(\|q - p_i(t)\|, \tau_i(t))$: Performance Function

$\phi(q)$: Density Function $\quad H_{ts}(\mathcal{P}, \mathcal{T})$: Distortion Function

$$H_{ts}(\mathcal{P}, \mathcal{T}) := \sum_{i=1}^{n} \int_0^T \int_{V_i^{ts}(\mathcal{P}, \mathcal{T})} f_{ts}(\|q - p_i(t)\|, \tau_i(t))\phi(q)dqdt$$

Asynchronous Coverage Control Algorithm

Tokyo Institute of Technology

Asynchronous Coverage Control Algorithm

Initial Conditions

For $i \in \{1, 2, \ldots, n\}$, let local time $t_i := 0$.
Command monitor() and get neighbors $\mathcal{N}_i(t_i)$. Send $p_i(t_i)$, $\tau_i(t_i)$ to neighbors $\mathcal{N}_i(t_i)$. Compute time-space Voronoi region V_i^{ts}, compute centroid $C_{V_i^{ts}}^{ts}$, compute projected centroid $C_{V_i^{ts}}^{s}$.

Asynchronous Coverage Control Algorithm

Tokyo Institute of Technology

Step 1

Command monitor() and get neighbors $\mathcal{N}_i(t_i)$. Send $p_i(t_i)$, $\tau_i(t_i)$ to neighbors $\mathcal{N}_i(t_i)$.
If $p_i(t_i) = C_{V_i^{ts}}^{s}$, compute time-space Voronoi region V_i^{ts}, compute centroid $C_{V_i^{ts}}^{ts}$, compute projected centroid $C_{V_i^{ts}}^{s}$, and send $stop_i$ to neighbors $\mathcal{N}_i(t_i)$.
If $stop_j$ is sent, compute only $i-j$ partition and update only $i-j$ partition in time-space Voronoi partitions, compute time-space Voronoi region V_i^{ts}, compute centroid $C_{V_i^{ts}}^{ts}$, compute projected centroid $C_{V_i^{ts}}^{s}$.

Asynchronous Coverage Control Algorithm

Tokyo Institute of Technology

Step 2

Compute velocity parameter k_i, compute $u_i(t_i) := k_i(C_{V_i^{ts}}^{s} - p_i(t_i))$, input $u_i(t_i)$.

Step 3

Let local time $t_i := t_i + 1$ and go to Step 1.

Function List
monitor(): searching neighbors utilizing MONITORING ALGORITHM.

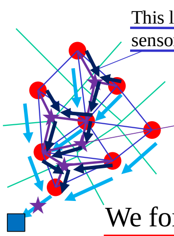
J. Cortes, S. Martinez, T. Karatas and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, Vol. 20, No. 2, pp. 243-255, 2004.

Outline

Tokyo Institute of Technology

Optimal Assignment for Routing

Tokyo Institute of Technology



This line connects between sensors in neighborhood.

Balancing and decreasing the power consumption of each agent.

We formulate an optimization problem.

: Sensor Agent
: Base Station
: Voronoi Region
: Relay Agent
: Delaunay Triangle

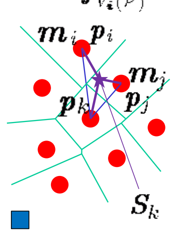
An Optimal Assignment Problem for Routing

Tokyo Institute of Technology

Cost of Transmission of a Single Data: $P(d) = a + bd^{\alpha}$

$a, b \in \mathbb{R}_+$ $\alpha \in \{2, 3, 4, 5, 6\}$ d : Distance

$m_i = \int_{V_i(\mathcal{P})} \phi(q) dq$ S_k : Total Cost of Transmission to k-th Sensor Agent



$$S_k = m_i P(\|p_i - p_k\|) + m_j P(\|p_j - p_k\|)$$

$S_k \Rightarrow S'_k$ **if** $S_k > S'_k$

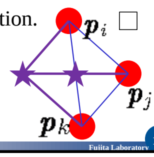
An Optimal Assignment Problem for Routing

Lemma

Relay agent that minimizes total cost of transmission to k-th sensor agent is in closure of Delaunay triangle.

Proof

If relay agent is not in closure of Delaunay triangle, there exists a position that its total cost of transmission is less than minimal total cost of transmission in closure of Delaunay triangle. This is a contradiction. □



An Optimal Assignment Problem for Routing

An Optimal Assignment Problem for Routing

$$\min_{r_1, r_2, \ldots, r_R} S(\mathcal{P}, \mathcal{R})$$

$S(\mathcal{P}, \mathcal{R})$: Total Cost of Transmission

$\mathcal{R} = \{r_1, r_2, \ldots, r_R\}$

r_l : Position of l-th Relay Agent

Outline

1. Introduction
2. Asynchronous Coverage Control
3. Optimal Assignment for Routing
4. Conclusions and Future Works

Conclusions and Future Works

Conclusions

- We formulated an asynchronous distortion problem.
- We proposed an asynchronous coverage control algorithm.
- We formulated an optimal assignment problem for routing.

Future Works

- Creating algorithms for above problems.

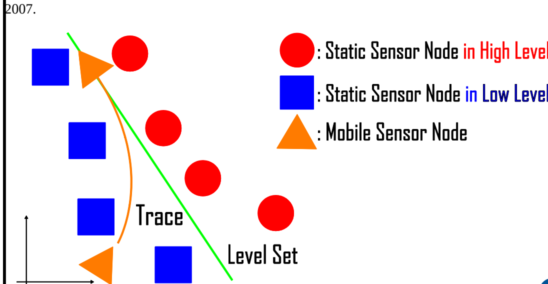
Appendix

1. Network-Based Control
2. Support Vector Machine
3. Statistical Learning Theory
4. Conclusions and Future Works

Network-Based Control

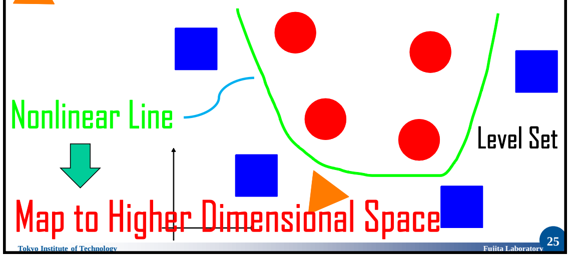
Level Sets of Scalar Fields

K. Dantu and G. S. Sukhatme, “Detecting and Tracking Level Sets of Scalar Fields using a Robotic Sensor Network,” *IEEE International Conference on Robotics and Automation*, pp. 3665-3672, 2007.

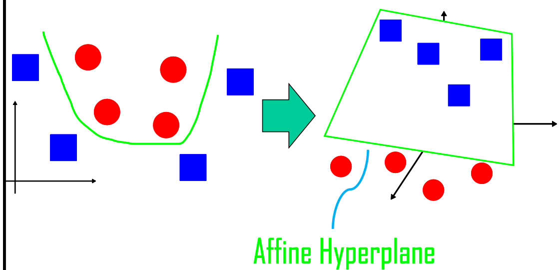


Network-Based Control

- : Static Sensor Node in High Level
- : Static Sensor Node in Low Level
- : Mobile Sensor Node



Network-Based Control



We utilize **Support Vector Machine**.

What is a Support Vector Machine (SVM)?

What is a Support Vector Machine (SVM)?

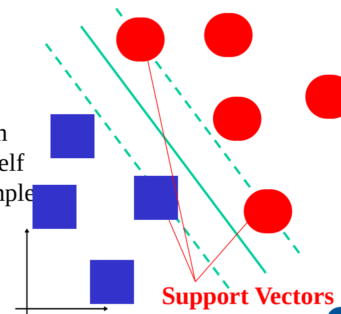
SVM is a learning machine developed for solving 2-class classification problem.

- finding an optimal separating hyperplane
- utilizing a soft-margin technique for inseparable data
- handling non-linear rules utilizing kernels
- no local optima

Linear SVM

What is an optimal separating hyperplane ?

A hyperplane that maximizes minimum distance between itself and any training sample of each class.



Linear SVM

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|_2^2$$

$$s.t.\quad y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1$$

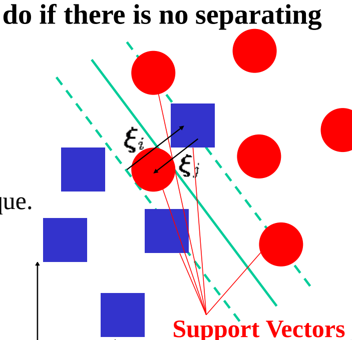
$$\Rightarrow \boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i$$

$\boldsymbol{x}_i \in \mathbb{R}^n$, $y_i \in \{1,-1\}$, $\mathbb{R}^n \ni \boldsymbol{w}$: weight vector, $\mathbb{R} \ni b$: bias term, $0 \leq \alpha_i$: Lagrange multiplier.

Soft-Margin SVM

What could we do if there is no separating hyperplane ?

We could utilize a soft-margin technique.



Soft-Margin SVM

$$\min_{\boldsymbol{w},b,\xi_i} \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i$$

$$s.t.\ \ y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1-\xi_i,\ \ \xi_i \geq 0$$

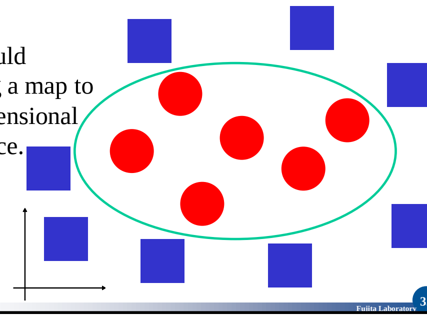
$$\Rightarrow\ \boldsymbol{w} = \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i$$

$\boldsymbol{x}_i \in \mathbb{R}^n$, $y_i \in \{1,-1\}$, $\mathbb{R}^n \ni \boldsymbol{w}$: weight vector,
$\mathbb{R} \ni b$: bias term, $0 \leq \alpha_i$: Lagrange multiplier,
$\mathbb{R} \ni C$: tuning parameter.

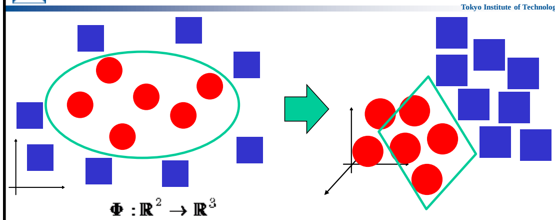
Non-linear SVM

Could we classify 2-class non-linearly ?

Yes, we could by utilizing a map to higher dimensional feature space.



Non-linear SVM



$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$

$$(x_1, x_2) \to (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

$$(\Phi(\boldsymbol{x})\cdot\Phi(\boldsymbol{y})) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)(y_1^2, \sqrt{2}y_1y_2, y_2^2)^\top$$

$$= ((x_1, x_2)(y_1, y_2)^\top)^2$$

$$= (\boldsymbol{x}\cdot\boldsymbol{y})^2$$

$$=: K(\boldsymbol{x},\boldsymbol{y})$$

Kernel Function

Non-linear SVM

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|_2^2$$

Map to Higher Dimensional Feature Space

$$s.t.\ \ y_i(\boldsymbol{w}^\top \Phi(\boldsymbol{x}_i) + b) \geq 1$$

$$\Rightarrow\ \boldsymbol{w} = \sum_{i=1}^{n}\alpha_i y_i \Phi(\boldsymbol{x}_i)$$

$\boldsymbol{x}_i \in \mathbb{R}^n$, $y_i \in \{1,-1\}$, $\mathbb{R}^n \ni \boldsymbol{w}$: weight vector,
$\mathbb{R} \ni b$: bias term, $0 \leq \alpha_i$: Lagrange multiplier,
$\Phi : \mathbb{R}^n \to \mathbb{R}^m$, $n < m$.

Kernel Method

$$\boldsymbol{w}^\top\Phi(\boldsymbol{x}_i) = \sum_{j=1}^{n}\alpha_j y_j \Phi^\top(\boldsymbol{x}_j)\Phi(\boldsymbol{x}_i)$$

Kernel Function

$$= \sum_{j=1}^{n}\alpha_j y_j K(\boldsymbol{x}_j, \boldsymbol{x}_i)$$

Kernel Trick

$\Rightarrow$ Calculation of inner product in higher dimensional feature space is replaced by that of kernel function.

e.g.

$$(\Phi(\boldsymbol{x})\cdot\Phi(\boldsymbol{y})) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)(y_1^2, \sqrt{2}y_1y_2, y_2^2)^\top$$

$$= ((x_1, x_2)(y_1, y_2)^\top)^2$$

$$= (\boldsymbol{x}\cdot\boldsymbol{y})^2$$

$$=: K(\boldsymbol{x},\boldsymbol{y})$$

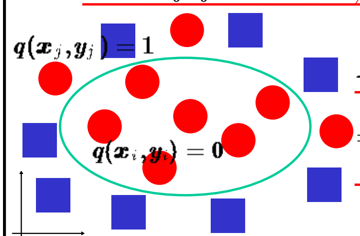
Kernel Function

Statistical Learning Theory

Loss Function : $q(\boldsymbol{x},y) = \begin{cases} 0 & \text{if}\ \ y(\boldsymbol{w}^\top\Phi(\boldsymbol{x})+b) \geq 0 \\ 1 & \text{if}\ \ y(\boldsymbol{w}^\top\Phi(\boldsymbol{x})+b) < 0 \end{cases}$

Risk : $R(f) = \int\int q(\boldsymbol{x},y)p(\boldsymbol{x},y)d\boldsymbol{x}dy$

No way to know!



$q(\boldsymbol{x}_j, y_j) = 1$

$q(\boldsymbol{x}_i, y_i) = 0$

Empirical Risk :

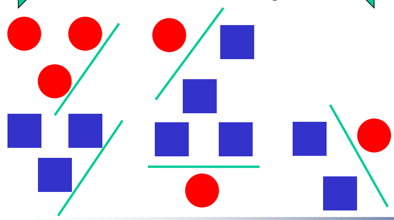
$$R_{emp}(f) = \frac{1}{n}\sum_{i=1}^{n} q(\boldsymbol{x}_i, y_i)$$

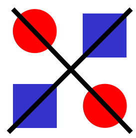
VC Dimension

VC dimension of a class $\mathcal{G}$ (a family of discriminant functions) is the size of the largest set that it can shatter.

e.g. $\mathcal{G}$ is the set of half-planes in $\mathbb{R}^2$.

VC dimension of a class $\mathcal{G}$ is 3.





Vapnik-Chervonenkis Theory

Theorem

For $\delta > 0$ with probability at least $1-\delta$,

$$\sup_{f \in \mathcal{F}} |R(f) - R_{emp}(f)| \le \frac{\varepsilon(n)}{2}\left(1+\sqrt{1+\frac{R_{emp}(f)}{\varepsilon(n)}}\right)$$

h : VC dimension of a class $\mathcal{F}$, $\varepsilon(n) = 4\dfrac{\ln\left(\frac{2ne}{h}\right)^h - \ln\left(\frac{\delta}{4}\right)}{n}$.

e.g.

$\delta = 0.01$

$h = 3$

$n = 100 \Rightarrow \varepsilon \approx 0.86$

$n = 1000 \Rightarrow \varepsilon \approx 0.11$

$n = 10000 \Rightarrow \varepsilon \approx 0.014$

We need so many sample data!

V. N. Vapnik, "Statistical Learning Theory," *John Wiley & Sons, New York*, 1998.

Conclusions and Future Works

Conclusions

- We studied SVM and Statistical Learning Theory.
- We found that we need so many sample data for utilizing SVM.

Future Works

- Finding a solution to above problem.