UNIVERSITY OF PADOVA
DEPARTMENT OF ENGINEERING AND MANAGEMENT

Multi-Agent Systems: Modeling, Estimation and Control Issues - Case Study: Camera Networks -

Angelo Cenedese

joint work with L.Schenato

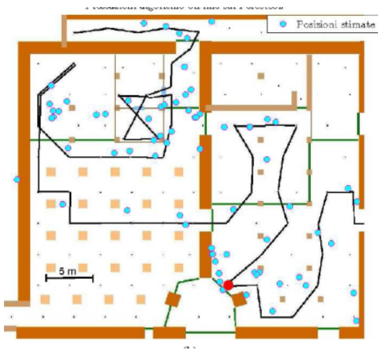
UCSB - 12nd May 2010

Outline of the Talk

- **Introduction and motivation**
- **Formalization**
- **The camera network**
  - **The graph building problem**
  - **The task assignment problem**
  - **The coordination problem**
- **Summary**

Motivation and Applications

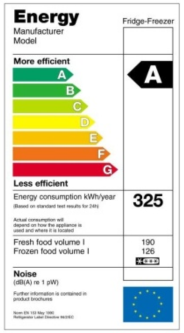
Localization & Tracking



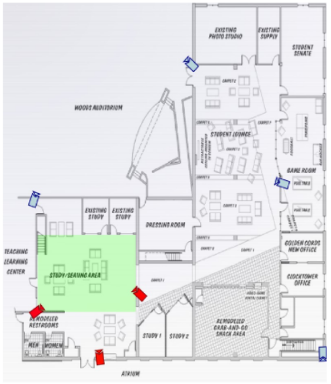
Swarm robotics

Smart Greenhouse

Energetic Auditing & Building Energy Management



Monitoring & Surveillance



Networked Controlled Systems

Sensor Network + Network Controlled Systems

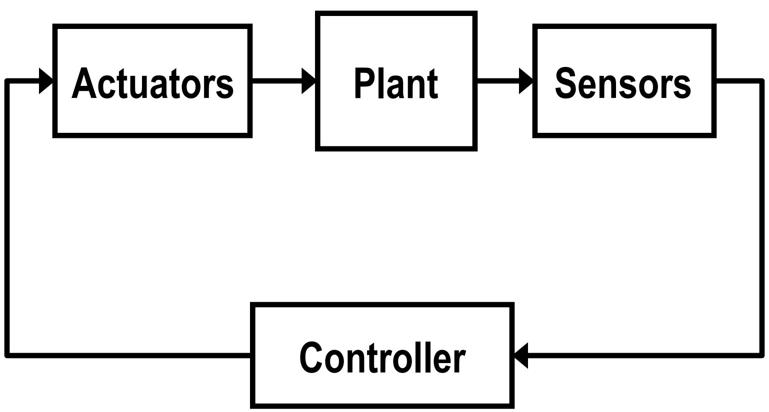
Sensor and Actor Network (SAN):

physically distributed dynamical systems interconnected by a communication network.

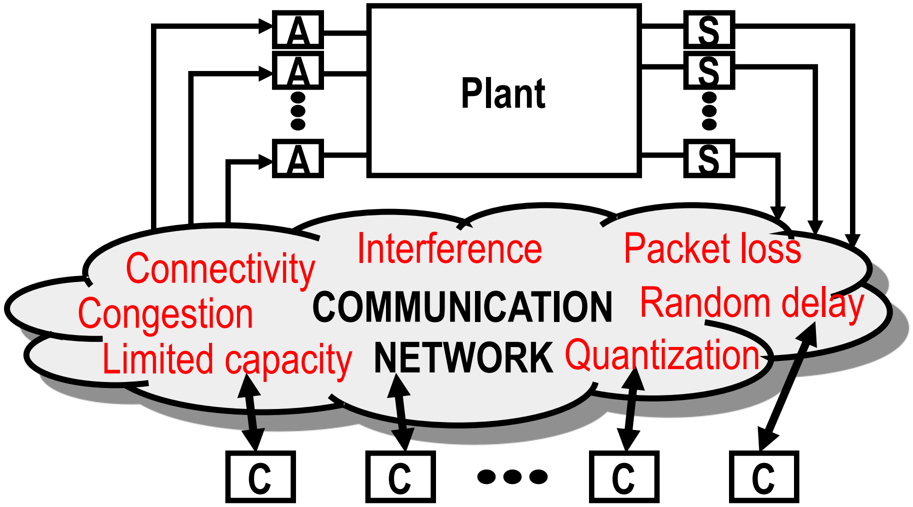
Nodes are sensors (*monitoring*) and/or actuators (*control*).

A new control paradigm:

Classical architecture:
Centralized structure



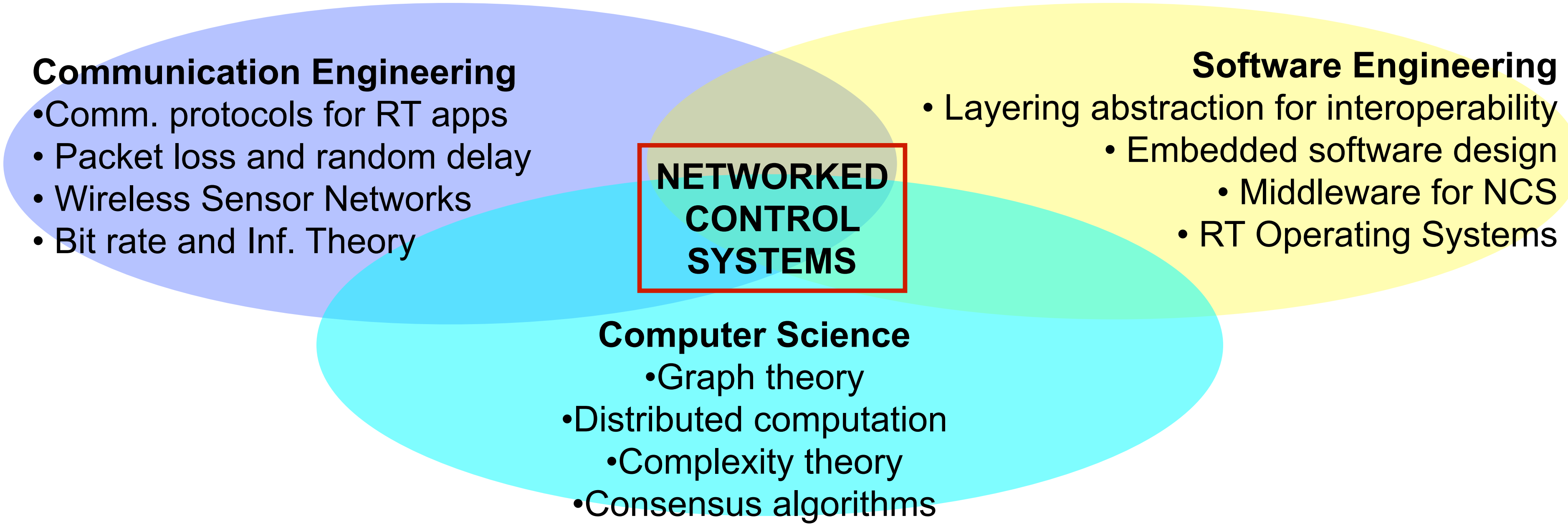
Large scale distributed structure



Common Features and Research Issues

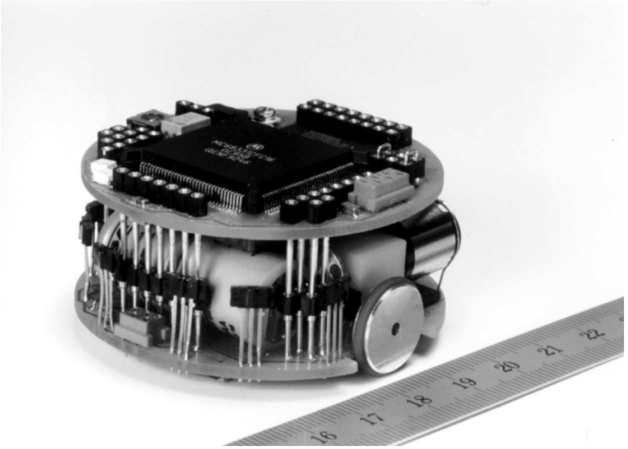
- Distributed estimation and control
- Control under packet loss & random delay
- Sensor fusion
- Distributed time synchronization
- Coordination & Cooperation
- Complex model identification
- Sensor selection for identification
- Optimal sensor placement

Interdisciplinary research needed



Agent Networks

- Usually small & cheap devices
- Computational/Control capabilities
- Communication & Memory
- Wired/Wireless communication
- Battery powered/Energy scavenging
- Sensors & Actuators



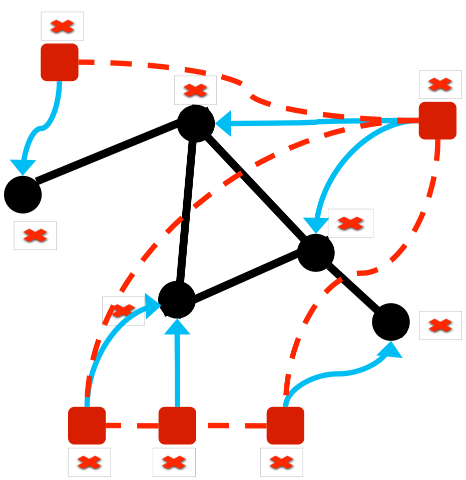
…but not only these…

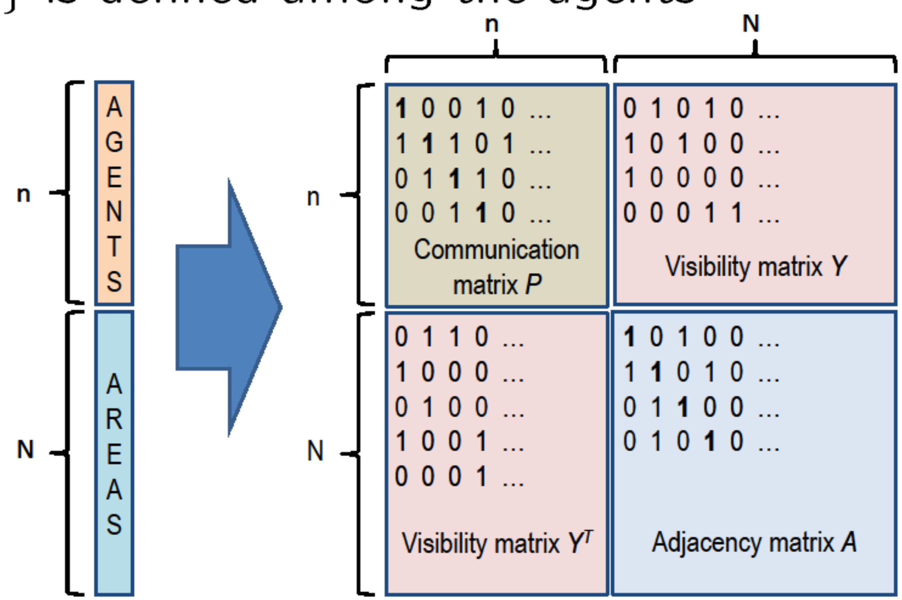
Outline of the Talk

- Introduction and motivation
- **Formalization**
- The camera network
  - The graph building problem
  - The task assignment problem
  - The coordination problem
- Summary

Network Abstraction 1/2

- The agent network is represented by a multidimensional graph:
  - the environment is organized into **areas**: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$.
    Vertices $\mathcal{V} = \{v_1, \ldots, v_N\}$ are the interesting locations to be monitored; **edges** $\mathcal{E}$ are the adjacency (directed) links among the areas;
  - a set of **agents** (sensors and/or actuators) is distributed in the environment: $\mathcal{A} = \{a_1, \ldots, a_n\}$;
  - each area is linked to a subset of agents: $\forall\, v_i \in \mathcal{V}$, a set is defined $\mathcal{A}(v_i) \subseteq \mathcal{A}$;
  - a communication graph $\mathcal{C} = \{\mathcal{A}, \mathcal{L}\}$ is defined among the agents





Network Abstraction 2/2

- Also:

  Finite Resource: each agent exploits a finite number of **resources** $\mathcal{R} = \{r_1, \ldots, r_m\}$: $\forall\, a_j \in \mathcal{A}$, a set is defined $\mathcal{R}(a_j) \subseteq \mathcal{R}$.

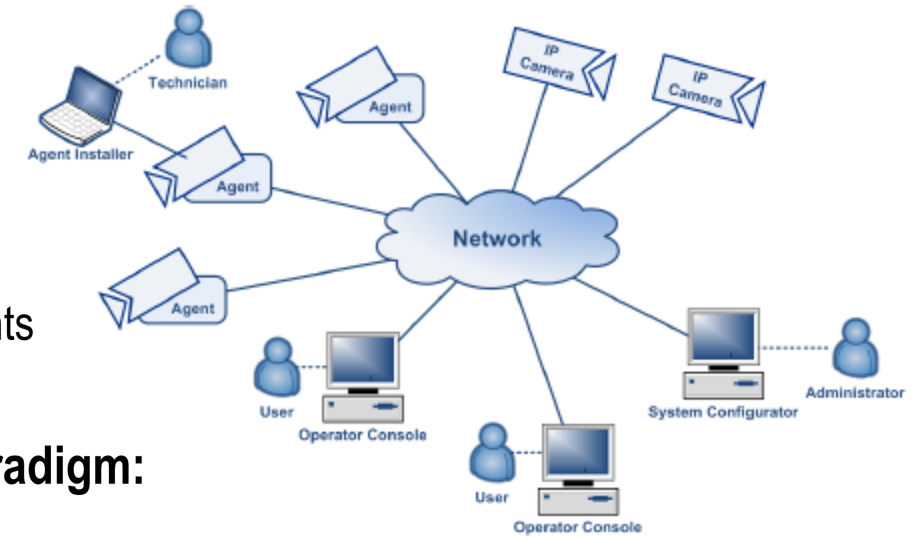
  Event Driven: a set of **events** is detected in the environment: $\mathcal{H} = \{h_1, \ldots, h_p\}$;

  Multi Task: a set of **tasks** is issued to the network: $\mathcal{T} = \{t_1, \ldots, t_s\}$.

Outline of the Talk

- Introduction and motivation
- Formalization
- **The camera network**
  - The graph building problem
  - The task assignment problem
  - The coordination problem
- Summary

Camera Networks for Videosurveillance



Paradigmatic Case study:

- High communication bitstream
- High sensitivity to performance
- Real-world scenarios with increasing number of agents

The Surveillance Network realizes the SAN paradigm:

- "Dynamic" sensing: the sensors can be actuated

and their parameters be tuned according to the dynamics of the scene

- Actuators: I/O signals to undertake RT actions

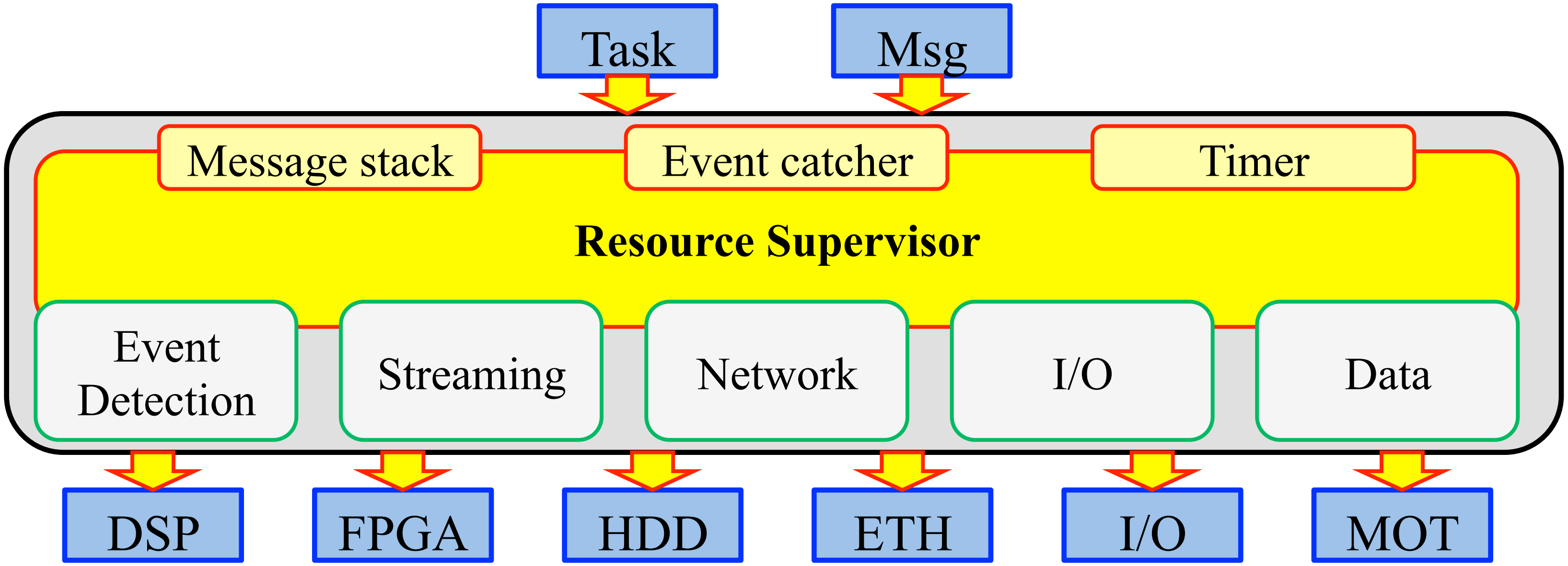
The network architecture should guarantee:

- Dynamic area coverage according to operational needs.
- Flexibility in terms of space reconfiguration and personalized environment
- Scalability/expandability in terms of adding nodes/adopting new technologies

The Smart Camera/Agent

The Finite Resource Agent:

- PTZ controller (nb: input and output)
- Streaming device (live and playback)
- Network controller
- I/O controller
- Event detection engine
- Mass storage controller



Distributed Control Issues

1. **Graph Building Problem**: build the graph structure $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, from the knowledge of $\mathcal{V}$ and a set of training data $\mathcal{M}$.

2. **Adaptive dynamic task allocation** with the finite resource constraint as a Fault Detection Problem or through a game theoretic approach.

3. **Agent coordination for distributed task**: Constrained Optimization Problem (minimization of number of agents or maximization of performance).

4. **Data Redundancy Problem**: self-configuration of redundancy policy based on local decisions, knowing the agent priority, the data throughput, the probability failure.

5. **Model identification**: estimation of the timescales of the system, of the delays, of the model uncertainties.

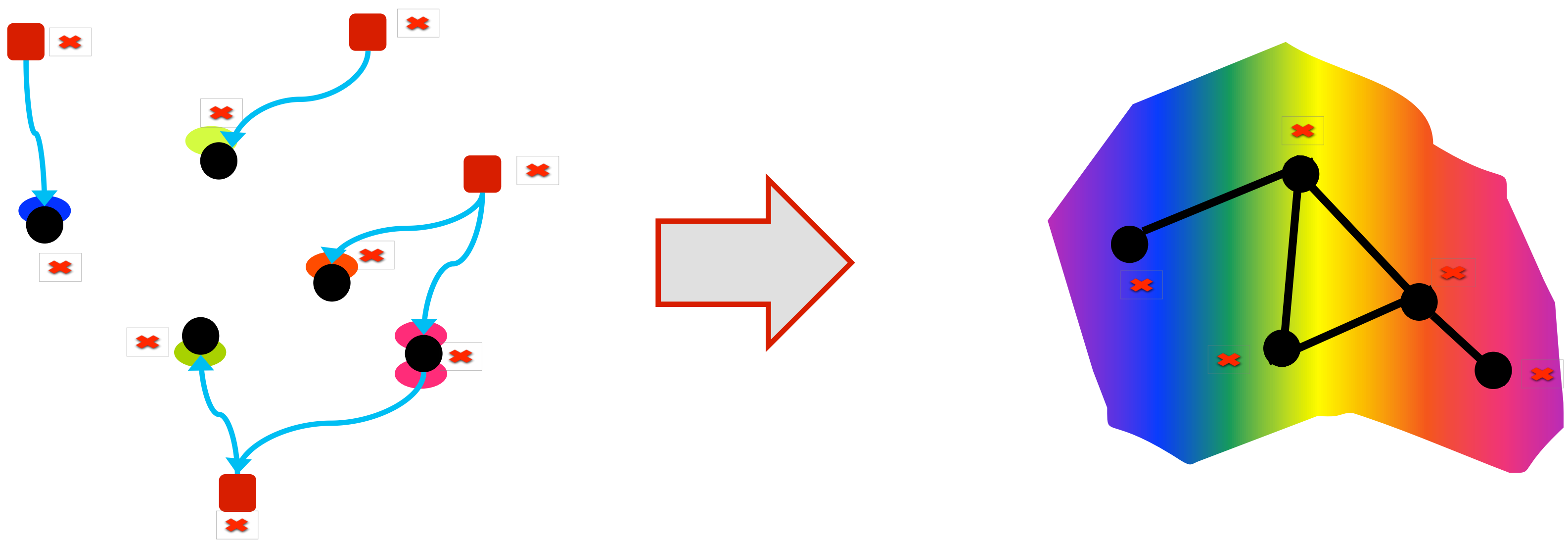
Outline of the Talk

- Introduction and motivation
- Formalization
- The camera network
  - **The graph building problem**
  - The task assignment problem
  - The coordination problem
- Summary

Rationale

- *Automatic configuration of the system: is it possible to build the graph structures from a set of training data?*

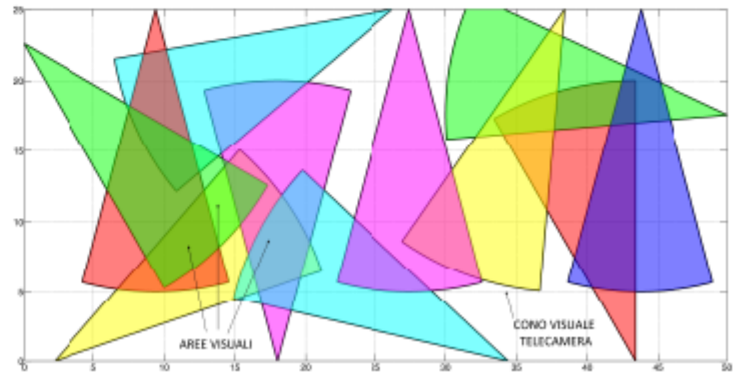
- Given the nodes $\mathcal{V}$ and $\mathcal{A}(v_i)$, and a set of measurements $\mathcal{O}(v_i, t)$ from local (temporal and spatial) data association:
  - infer the **adjacency** links $\mathcal{E}$ to build the graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$: high probability path reconstruction problem from data;



Case Study Scenario

Simplified scenario:

- 2D domain Ω of $25 \times 50m^2$
- $K = 11$ fixed cameras $\{\mathcal{A}_i\}$ positioned around the perimeter
- $\{\mathcal{A}_i\}$ with circular sector field of view (fov) Δ_i



Unknowns:

- camera positions and orientations
- shape of the fov overlapping regions

Assumptions:

- States of the system are the visible areas (overlapping and non overlapping fov's) + (null) state corresponding to no observation and region $S_0 = (\Omega \setminus \bigcup \Delta_i)$
- Observation at time t is given as a binary string $O_t \in \{0,1\}^K$ whose entry in the i-th position is 1 if the i-th camera sees the target object, while it is 0 otherwise

Statement of the Problem

***Note:**

We are not interested in solving the computational vision problem of how *the camera sees the object of interest, but we are interested in* if *the camera sees it*

Problem:

Given an observation sequence $\mathcal{O}$ in the finite interval $[1, T]$, infer the set of states $\mathcal{S}$ and the underlying graph $\mathcal{G}$ that constraints their transitions (and whose node set is $\mathcal{S}$).

Solution:

Two step approach:

1. a strong correspondence between states and (static) observations is enforced and for every different observation at time t a state is generated and a transition probability is evaluated
2. some of these states undergo a splitting procedure, giving rise to the revelation of initially hidden states, in this ways being replaced by two or more novel states

Hidden Markov Model (HMM)

- HMM is the model that better describes the problem: the state is not known a priori but need to be inferred from observations

- HMM characterization:
  - N-state set is $\mathcal{S} = \{S_1, S_2, \dots, S_N\}$ and state at time t is $q_t \in \mathcal{S}$
  - M distinct observation symbols, $M \leq 2^K$: $\mathcal{V} = \{v_1, v_2, \dots, v_M\}$, $v_i \in \{0, 1\}^K$

  a sequence of observations in $\mathcal{T} = [1, T] \subset \mathbb{Z}_+$ is $\mathcal{O} = [O_1, \dots, O_T]$, $O_i \in \mathcal{V}$
  - state transition probability distribution $A \in \mathbb{R}^{N \times N}$:

$$a_{ij} = \mathbf{P}[q_{t+1} = S_j | q_t = S_i] \qquad 1 \leq i, j \leq N$$

  - observation symbol probability distribution $B \in \mathbb{R}^{M \times N}$:

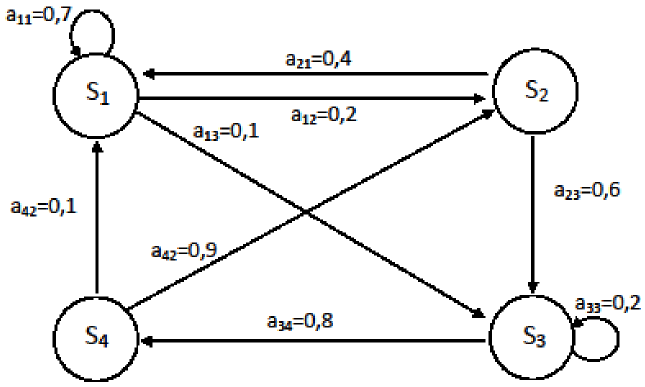
$$b_j(v_i) = \mathbf{P}[O_t = v_i | q_t = S_j] \qquad 1 \leq i \leq M\,, \quad 1 \leq j \leq N\,,$$

  - the initial state distribution $\pi \in \mathbb{R}^N$:

$$\pi_i = \mathbf{P}[q_1 = S_i] \qquad 1 \leq i \leq N\,,$$

HMM Associated Graph

- Given the state space $\mathcal{S}$, the Markov model is the triple $\lambda = (A, B, \pi)$, and a directed graph $\mathcal{G}$ is associated to the related Markov chain.
- Event $[q_t = S_i]$ represents the fact that the chain is in the state S_i at time t, while the event $[q_t = S_j | q_{t-1} = S_i]$ represents the transition from the state S_i to the state S_j.
- $\mathcal{G}$ is obtained as follows:
  1. for every state S_i in the model a graph node is set;
  2. for each pair of nodes (S_i, S_j) in the graph, the directed arc between the two is labeled with the transition probability a_{ij}.



The Forward-Backward Algorithm

Given an initial model, $\lambda_0 = (A, B, \pi)_0$, adjust the model parameters $\{A, B, \pi\}$ to maximize the probability of the observation sequence $\mathcal{O}$ given the model:

$$\max_{\lambda} \mathbf{P}[\mathcal{O}|\lambda]$$

Be the observation interval $\mathcal{T} = [1, T]$:

- **forward variable** $\alpha_t(i) = \mathbf{P}[O_1, O_2, \ldots, O_t, q_t = S_i|\lambda]$:
probability of the partial observation sequence in $[1, t]$ and the state, given the model
- **backward variable** $\beta_t(i) = \mathbf{P}[O_{t+1}, O_{t+2}, \ldots, O_T|q_t = S_i, \lambda]$:
probability of the partial observation sequence in $[t + 1, T]$, given the state and the model
- **state probability variable** $\gamma_t(i) = \mathbf{P}[q_t = S_i|\mathcal{O}, \lambda]$:
probability of being in the state given the observation sequence and the model.

The probability of the whole observation sequence $\mathcal{O}$ and of being in the state S_i is given in terms of forward and backward variables:

$$\mathbf{P}[\mathcal{O}|\lambda] = \sum_{i=1}^{N} \alpha_T(i) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \qquad \gamma_t(i) = \mathbf{P}[q_t = S_i|\mathcal{O}, \lambda] = \frac{\alpha_t(i)\beta_t(i)}{\mathbf{P}[\mathcal{O}|\lambda]}.$$

The Baum-Welch Algorithm

Baum-Welch algorithm: expectation-maximization method to compute posterior estimates for the model parameters, based on a maximum likelihood approach.

1. compute the forward and backward probabilities for each state of the model:

$$\xi_t(i,j) = \mathbf{P}[q_t = S_i, q_{t+1} = S_j | \mathcal{O}, \lambda] = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$$

2. evaluate the expected count of the transition to a state and emission of an observation pair, to provide a new estimation of $\{\bar{A}, \bar{B}, \bar{\pi}\}$:

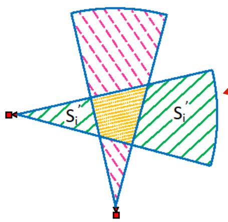
$$\bar{\pi}_i = \mathbb{E}[\# \text{ of } S_i \text{ at } t = 1]$$

$$\bar{a}_{ij} = \frac{\mathbb{E}[\# \text{ of } S_i \to S_j]}{\mathbb{E}[\# \text{ of } S_i \to \bullet]} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \qquad \bar{b}_j(v_i) = \frac{\mathbb{E}[\# \text{ of } S_j \text{ and } v_i]}{\mathbb{E}[\# \text{ of } S_j]} = \frac{\sum_{\substack{t=1 \\ \text{s.t. } O_t = v_i}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

The new model $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ is more likely than the original λ to produce the observation sequence (or at least equally likely):

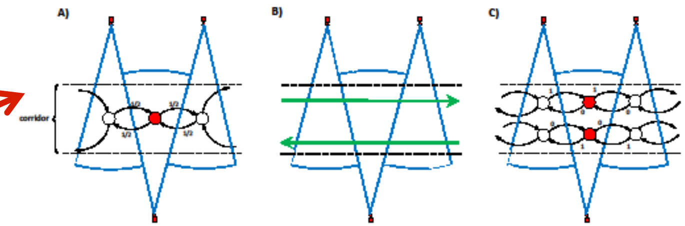
$$\mathbf{P}[\mathcal{O}|\bar{\lambda}] \geq \mathbf{P}[\mathcal{O}|\lambda)]$$

The Splitting procedure



Topological Splitting

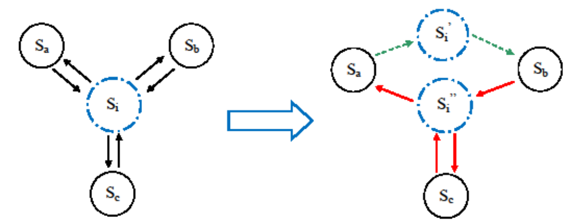
Logical Splitting



- A table is built with the possible past and future nodes (q_{t-1} and q_{t+1}), and the related transition probabilities:

		q_{i+1}		
		S_a	S_b	S_c
	S_a	0	$\mathbf{P}_{a*b}$	0
q_{i-1}	S_b	$\mathbf{P}_{b*a}$	0	$\mathbf{P}_{b*c}$
	S_c	$\mathbf{P}_{c*a}$	0	$\mathbf{P}_{c*c}$

- **Projection:** the first row of the table is orthogonal to the others, which conversely lie on the same 2D space, and are in this sense "similar". The state S_i is split into two separate states S_i' and S_i'': S_i' will refer to the transition $S_a \rightarrow S_i \rightarrow S_b$ while S_i'' will be related to $\{S_b, S_c\} \rightarrow S_i \rightarrow \{S_a, S_c\}$.



- In general, a "measure of orthogonality" between row vectors r and s is given by the cosine of the angle between the two $\sigma_{rs} = \frac{r \cdot s^T}{\|r\|\|s\|}$

The Algorithm

Require: Set $\mathcal{S} = \mathcal{V}, B = I$, run Baum-Welch's algorithm

for all states S_i **do**

  while $\sigma_{hl} = 0$ for $a_{hi} > 0, a_{il} > 0$ **do**

    split node S_i into S_i and S_{N+1}

    set new $N+1$ row B equal to row i

    set new $N+1$ row and column of A

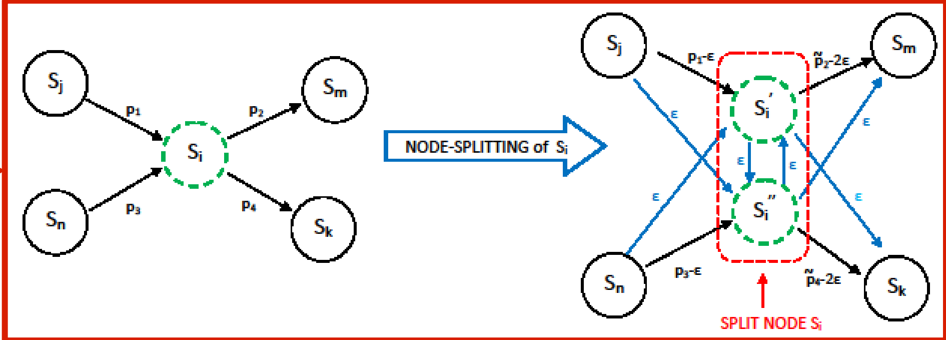
    repeat

      Baum-Welch's algorithm

    until convergence for A is not reached (w.r.t. ν)

  end while

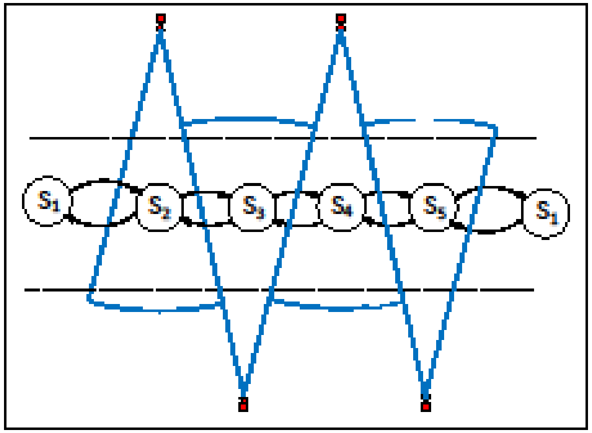
end for

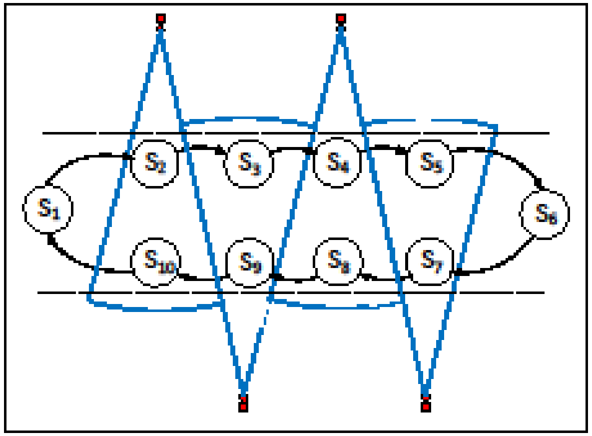


Index of performance:

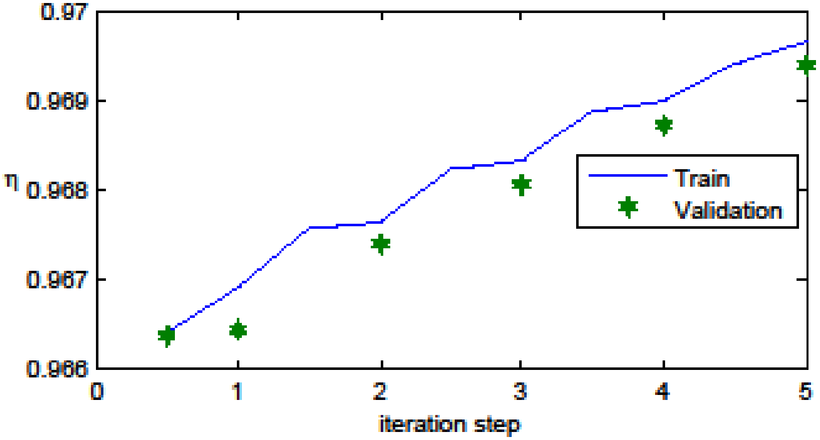
$$\eta(\mathcal{O},\lambda) = \sqrt[T]{\prod_{t=0}^{T-1} \mathbf{P}[\hat{y}_{t+1} = y_{t+1} | y_{0:t}]} = \sqrt[T]{\mathbf{P}[\mathcal{O}|\lambda]}$$

Simulations 1/2

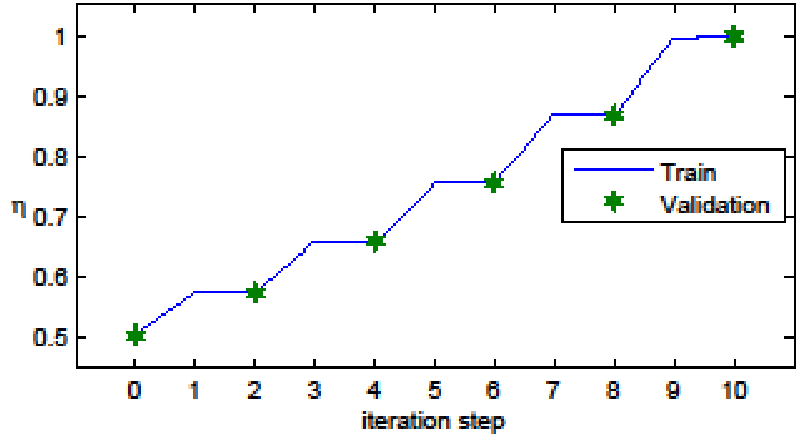




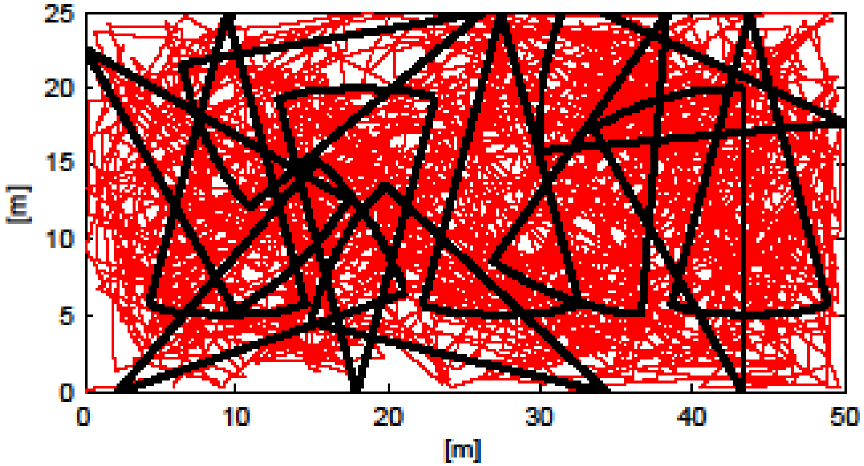
Self-loop allowed

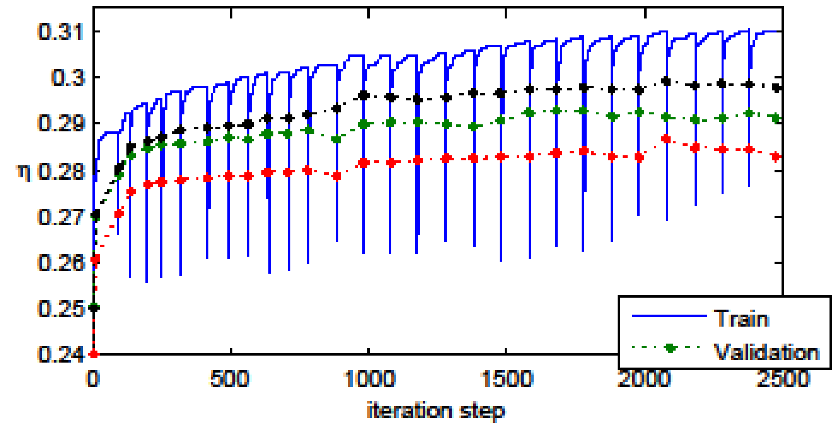


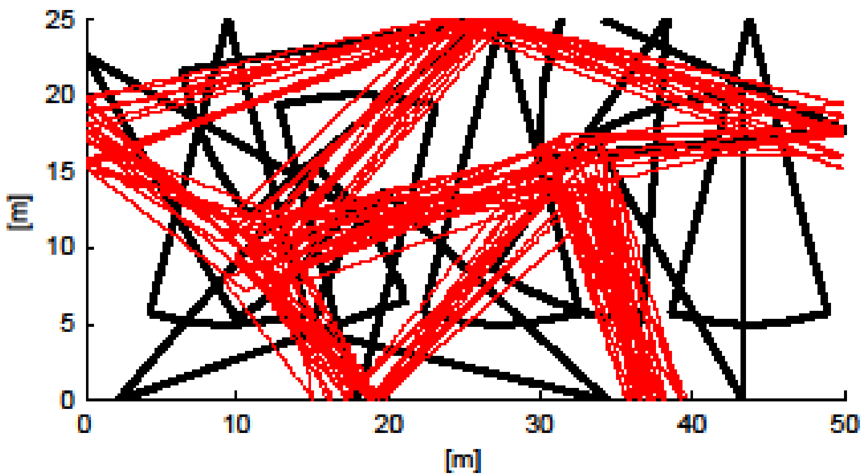
No self-loop

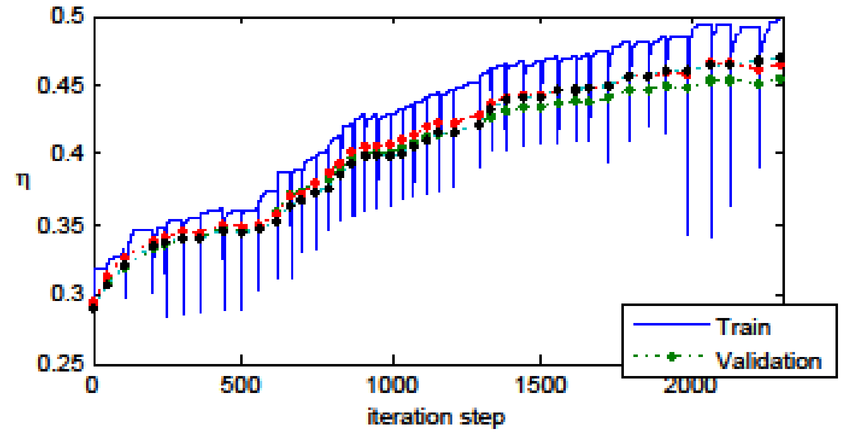


Simulations 2/2









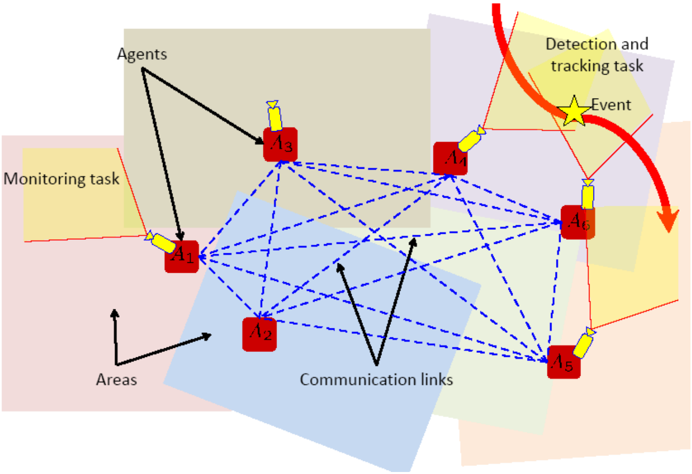
A.Cenedese , R.Ghirardello, R.Guiotto, F.Paggiaro , L.Schenato ,
On the Graph Building Problem in Camera Networks, submitted to NECSYS10

Outline of the Talk

- Introduction and motivation
- Formalization
- The camera network
  - The graph building problem
  - **The task assignment problem**
  - The coordination problem
- Summary

Rationale & Assumptions

- n agents $\mathcal{A} = \{a_1, \ldots, a_n\}$ in the environment, interconnected by a meshed communication graph. Each agent is capable of autonomous decision based on the complete information



- p areas of interest (field of views of at least one camera): a binary coverage matrix $V \in \mathbb{R}^{n \times p}$ is defined

- s different asynchronous task functions : $\mathcal{T} = \{\tau_1(\cdot), \ldots, \tau_s(\cdot)\}$

- List of agents that can fulfill task τ_j, $\mathcal{A}(\tau_j) \subseteq \mathcal{A}$; list of possible task for a_i, $\mathcal{T}(a_i)$

- A case study task set is considered:
  - automatic tracking $\tau_1(\cdot)$: area related task; medium priority;
  - manual tracking $\tau_2(\cdot)$: agent related task; high priority;
  - patroling task $\tau_3(\cdot)$: area related task; default task (zero priority);
  - playback streaming $\tau_4(\cdot)$: area related task; low priority.

Modeling 1/2

- **Data structures** for task management:
  Waiting Task List (WTL) – Active Task List (ATL) – **task pool** as $ATL \cup WTL$

- **Problem statement:** At a fixed time t, given the set of n agents and the presence of m_t task instances, to solve the task assignment problem means to find a binary array $\mathbf{x}_t \in \mathbb{R}^{nm_t}$ that maximizes some utility function $J(\mathbf{x}) : \mathbb{R}^{nm_t} \to \mathbb{R}$, with respect to the constraint $A_t\mathbf{x} \leq \mathrm{b}$.
  - $\mathbf{x}_t = [x_{1\,1}\, x_{1\,2}\, \ldots\, x_{1\,m_t}\, \ldots\, x_{n1}\, x_{n\,2}\, \ldots\, x_{n\,m_t}]^\top$: $x_{ij} = 1$ if the j-th task is given to the i-th agent, $x_{ij} = 0$ otherwise.
  - The binary matrix $A_t \in \mathbb{R}^{* \times nm_t}$: as many rows as the number of constraints:

    1: agents activity (e.g. one task per agent): $\sum_{j=1}^{m_t} x_{ij} \leq 1 \quad \forall i = 1, \ldots, n$

    2: task fulfilment (e.g. one agent per task): $\sum_{i=1}^{n} x_{ij} \leq 1 \quad \forall j = 1, \ldots, m_t$

    3: assignment feasibility (e.g. coverage matrix): $\sum_{i=1}^{n} \bar{v}_{ih} x_{ih} = 0 \quad \forall h = 1, \ldots, p$

    being $\bar{v}_{ih}$ the one's complement of the V-matrix entry $\bar{v}_{ih}$.

Modeling 2/2

- – By accounting only for Eq.1-2, the A_t matrix is of size $(n+m_t)\times(nm_t)$ can be therefore written as

$$A_t = \left[\begin{array}{c|c|c|c} \mathbf{1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & \dots & \mathbf{1} \\ \hline I_{m_t} & I_{m_t} & \dots & I_{m_t} \end{array}\right],$$

  where $\mathbf{0}$ and $\mathbf{1}$ are m_t-dimensional row vectors of zeros and ones respectively, and I_{m_t} is clearly the $m_t \times m_t$ identity matrix.

  – A_t is **totally unimodular** (TU): the integer constraints on the solution $x_{ij} \in \{0,1\}$ can be relaxed to $x_{ij} \in [0,1]$: $\mathbf{x}$ can be defined as a continuous variable.

  – The constraints Eq.3 in the A_t matrix formulation would lead to a non TU matrix, thus leading to a complex combinatorial problem and are implicitly taken into account by properly defining the utility function to be maximized.

- The **utility function** is a linear combination of the variable terms x_{ij}:

$$J(\mathbf{x},t) = \mathbf{c}(t)^\top \mathbf{x} = \sum_{i=1}^{n} \sum_{j=1}^{m_t} c_j(t) x_{ij},$$

  where if a particular task-agent assignment is feasible, for example w.r.t. the information given by the coverage matrix, the weight $c_j(t)$ is set to a positive value related to the priority of the task, otherwise it is set to $-\infty$.

The State Space 1/2

- A feasible solution vector $\mathbf{x}$ is chosen among $z_t = 2^{nm_t}$ different strings $\mathcal{Z}_t$
- The cardinality of the feasible solution set $\mathcal{X}_t \subset \mathcal{Z}_t$ is given by

$$|\mathcal{X}_t| = \sum_{k=0}^{\min(n,m_t)} \binom{n}{k} k! \binom{m_t}{k},$$

being k the number of allocated agents/undertaken tasks.

- If a default task is assumed for all agents (e.g. the patroling task), then at any instant no agent can remain unassigned in the network, and the cardinality of $\mathcal{X}_t$ is simply

$$|\mathcal{X}_t| = \min(n, m_t)! \binom{\max(n, m_t)}{\min(n, m_t)}$$

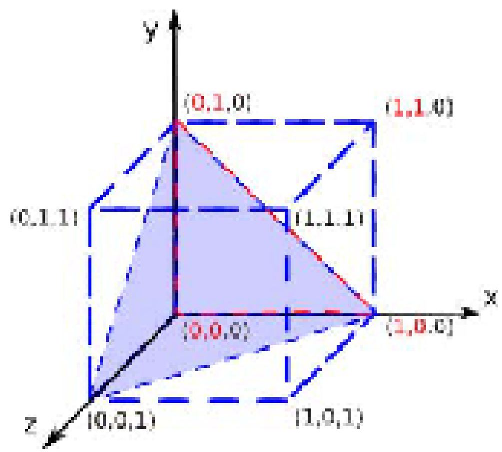
The problem is perfectly symmetric in the agent-task duality!

The State Space 2/2

Dynamic Task List:

A view of the set $\mathcal{Z}_t$ is given by the vertices of the unitary hypercube in $\mathbb{R}^{nm_t}$, while the feasible solution set $\mathcal{X}_t$ is obtained by intersecting the hypercube with the constraint hyperplane.

When a new task instance occurs, the solution space dimension increases to $\mathbb{R}^{n(m_t+1)}$ and the previous solution set $\mathcal{X}_t$ remains feasible also for the new task configuration although with the new task still unassigned: hence, in general, this solution may not be optimal because very different scenarios can take place.



The Stable Marriage Problem

Stable Marriage Problem (SMP):

- N men and N women, each with a (strict) preference list
- A marriage is **stable** if there is no pair of a man and a woman who both prefer another partner to their current one
- Gale-Shapley Algorithm:
  - based on men preference lists
  - attained marriage is stable and optimum for men: they are paired with their highest preferred woman among the possible stable solutions

Stable Marriage Problem with Ties and Incomplete Lists (SMTI):

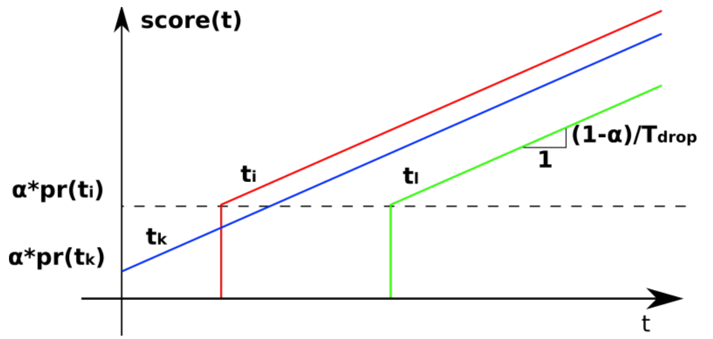
- Incomplete lists and/or not strict order
- A marriage $\mathcal{M}$ is **weakly stable** if there is no pair, each of whom is either unmatched in $\mathcal{M}$ and the other appears in his/her list, or strictly prefers the other to his/her partner in $\mathcal{M}$
- Combinatorial NP-hard problem: arbitrarily fixed tie breaking to converge.

The Stable Marriage Problem Revised

Task Dynamics:

- Each a_i gives every $\tau_j \in \mathcal{T}(a_i)$ a profit score:

$$c_j(t) = \alpha \cdot \mathrm{pr}(j) + \frac{1-\alpha}{T_{drop}} \cdot (t - T_{occ}(j))$$



- In between a new task arrival or task completion, the different $c_j(t)$'s grow with the same rate: the optimal assignment does not vary between these events and the $c_j(t)$'s can be evaluated only when a new task appear or an old task is completed.

- The a_i's preference list contains all the elements $\tau_j \in \mathcal{T}(a_i)$, ordered by c_j's.

Swapping Policy:

- Strategy to dynamically favor the assignment of tasks and avoid idle agents:

- Swap policy based on women (tasks) preference list:
  - number of tasks that follow τ_j in a_i's preference list
  - length of the a_i's preference list

- No optimality guarantee but shows stability, termination, good performance

Performance Metrics

- **Optimality:**

$$P_T(\mathbf{x}_t) = \frac{1}{T}\int_0^T \frac{\sum_{\tau_i \in ATL} \mathrm{pr}(\tau_i)}{\max_j \sum_n \mathrm{pr}(\tau_j)} dt$$

$\max_j \sum_n \mathrm{pr}(\tau_j)$ is the sum of the n higher intrinsic priority values in $ATL \cup WTL$;

- **idle state avoidance:**

$$I_T(\mathbf{x}_t) = \frac{1}{T}\int_0^T \sum_{i=1}^{n} \overline{\left(\sum_{j=1}^{m_t} x_{ij}\right)} \, dt$$

where the bar indicates the one's complement.

- **assignment interruption:**

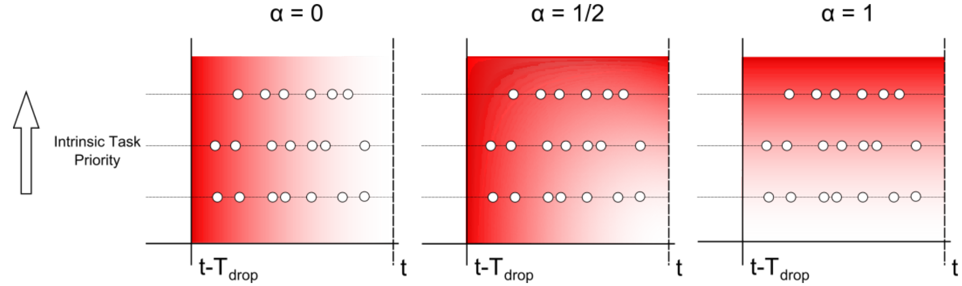
$$D_T(\mathbf{x}_t) = \frac{1}{T}\int_0^T \sum_{i=1}^{n} d_i(\mathbf{x}_t) \, dt$$

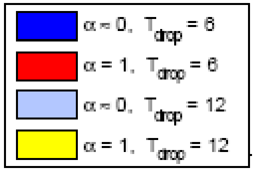
$d_i(\mathbf{x}_t)$ is an indicator function counting the interruptions;

- **average waiting time**
- **dropping rate**

Simulations

$$c_j(t) = \alpha \cdot \mathrm{pr}(j) + \frac{1-\alpha}{T_{drop}} \cdot (t - T_{occ}(j))$$





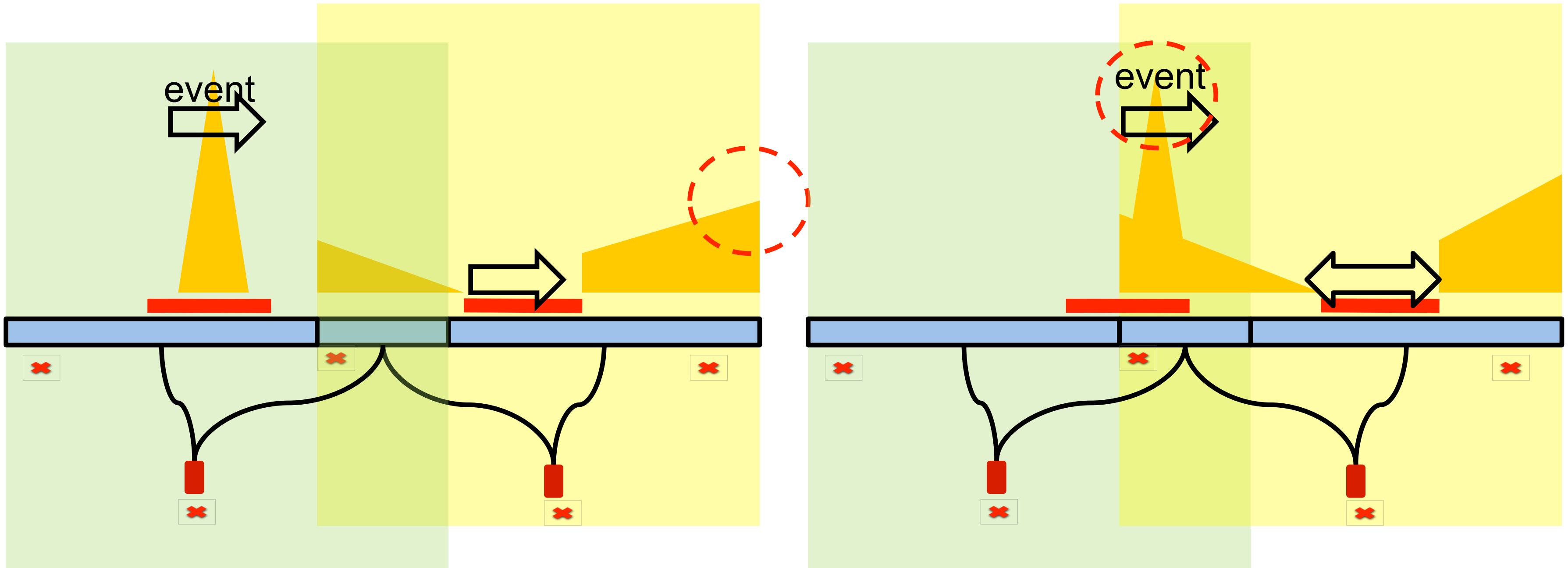
A.Cenedese, F.Cerruti, M.Fabbro, C.Masiero, L.Schenato,
Decentralized Task Assignment in Camera Networks, submitted to CDC10

Outline of the Talk

- Introduction and motivation
- Formalization
- The camera network
  - The graph building problem
  - The task assignment problem
  - **The coordination problem**
- Summary

The Coordination Problem

- A very simple example: Agent coordination for task completion:



Local decision functional:

$$J_{a_i}\left(t, v(a_i), v(a_i) \cup v(a_j), h(a_i), h(a_j), \forall a_j \in \mathcal{C}(a_i, v(a_i))\right)$$

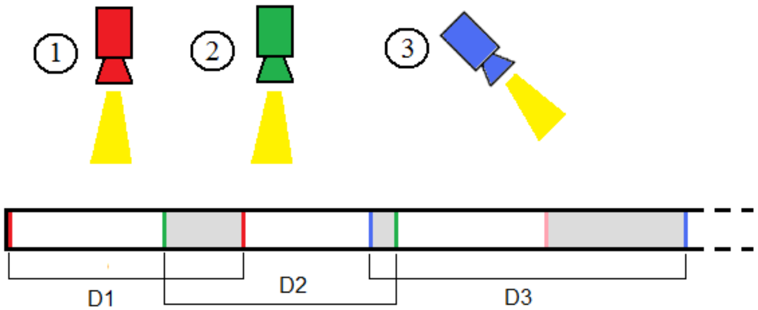
Global performance functional:

$$J(\mathcal{V}(\emptyset), \mathcal{H}(\emptyset))$$

Problem Formulation

Assumptions:

- 1-D scenario case
- 1-d.o.f. cameras: the camera f.o.v. changes due to pan movements only
- fixed coverage range: no alteration induced by view perspective



Notation:

- $L = [0, L_{tot}] \subset \mathbb{R}^+$: rectified total length of the monitored perimeter
- camera set $\{\mathcal{A}_1, \ldots, \mathcal{A}_N\}$
- $D_i = [D_{i,inf}, D_{i,sup}] \subset L$: total coverage range of i-th camera $\mathcal{A}_i$
- $v_i \in [-V_{i,max}, +V_{i,max}]$ is the (bounded) speed of i-th camera during pan movements
- $z_i(t) : \mathbb{R}^+ \rightarrow D_i$, $z_i(t) \in \mathcal{C}(\mathbb{R}^+)$: center of the area covered by the i-th camera
- $A_i = [a_{i-1}, a_i]$ is the steady state coverage of the i-th camera.

A feasible solution is such if $A_i \subseteq D_i \, \forall i$ and, if no overlapping zones are present, $\sum_{i=1}^{N} |A_i| = L_{tot}$, being $|A_i|$ the length of segment A_i.

Optimization Problem

Aim: find an optimal coverage criterion trading off between patroling and tracking

- **Functional** J**:** at each time instant t and position $x \in L$, $J = 0$ if location x is currently seen (by any camera), else it takes a positive real value as increasing as the time is passing since the last visit of x.

$$J(x,t,z(t)) : L \times \mathbb{R}^+ \times \mathcal{C}^N(\mathbb{R}^+) \rightarrow \mathbb{R}^+ \Rightarrow J(x,t,z(t)) = g(\bar{t}(x)) = 1 - e^{-\lambda \bar{t}(x)},$$

  where $\bar{t}(x) : L \rightarrow \mathbb{R}^+$, is the elapsed time from the last t s.t. $\exists i \in \{1, \dots, N\} | z_i(t) = x$

  The initial conditions ($t = 0$) are:

  - $z_i(0) \sim \mathcal{U}(D_i)$, uniformly distributed random variables in the interval D_i;
  - $\bar{t}(x) = 0$, $\forall x \in L$,

- **Optimization:** Minimization of J constrained to the system dynamics

$$\dot{z}_i(t) = v_i(t), \quad \forall\, i \quad s.t. \begin{cases} v_i(t) \in [-V_{i,max}, +V_{i,max}] \\ z_i(t) \in D_i \end{cases}$$

  and the speed set $\{v_i\}$ appears as a natural control input for the system:

$$V(t) = [\, v_1(t) \, \dots \, v_N(t) \,]', \quad t \in [0 + \infty).$$

Optimal Solution 1/2

Optimal solution without constraints:

Lemma: $N = 1$ camera $\mathcal{A}$ monitoring a perimeter L_{tot}: optimal solution by commanding a periodic motion with period $\bar{T}$ at the maximum speed:

$$\bar{V}(t) = V_{opt} = \pm V_{max} \qquad \bar{T} = \frac{2L_{tot}}{V_{max}} \qquad \bar{J} = J_{min} = 1 - e^{-\lambda \bar{T}}$$

The problem boils down to the selection of optimal coverage area $\{A_i\}$

Theorem: Optimal coverage is attained assuming that every camera is moving at its maximum speed $|V_{i,max}|$ with a periodical motion of period $\bar{T}$ in non-overlapping coverage areas A_i:

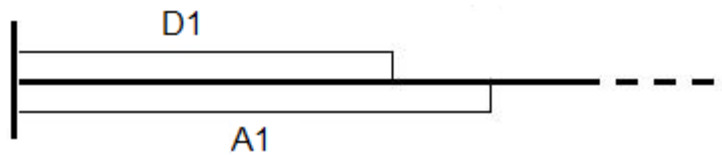
$$\bar{T} = 2T_o = \frac{2L_{tot}}{\sum_{i=1}^{N} V_{i,max}} \qquad |A_i| = V_{i,max} T_o$$

Optimal Solution 2/2

Optimal solution with constraints:

The optimal solution without any constraint is equivalent to the constrained optimal solution only if the unconstrained solution is feasible, that is if and only if

$$A_i \subseteq D_i$$



Theorem: If the unconstrained solution yields ($A_1 \not\subseteq D_1$), the optimal coverage of the trajectory is attained by assigning to $\mathcal{A}_1$ the maximum feasible length complying to its constraints, with

$$T_{o,c} = \frac{L_{tot} - |D_1|}{\sum_{i=2}^{N} V_{i,max}}, \qquad T_{o,c} > T_o$$

and recomputing the optimal solution for the remaining $N-1$ cameras to cover $L \setminus D_1$:

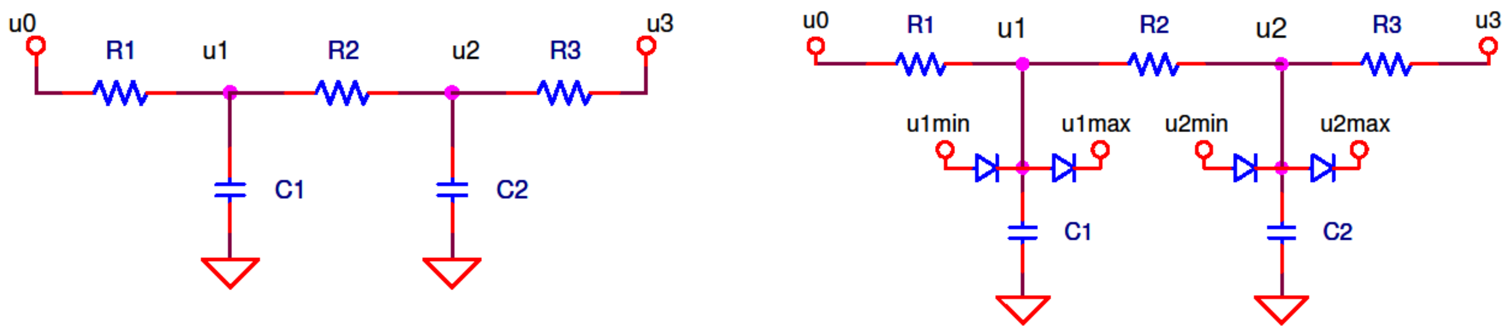
1. $A_1 = D_1$
2. $|A_i| = V_{i,max} T_{o,c} \quad i \neq 1$

The Distributed Model 1/2

Aim: find a method to reach the optimal steady-state configuration for patrolling extremes using only local interaction between neighboring cameras.

Electrical analogy:

relate voltages at circuit nodes to optimal patrolling sections for surveillance, and see resistor values as proportional to maximum patrolling speed of cameras.



Asymptotically stable equilibrium point for any initial configuration $\{u_1(0), \ldots, u_{N-1}(0)\}$.

Lyapunov function: $U = [u_0 \; u_1 \; \ldots \; u_N]^\top$

$$W(U(t)) = \sum_{i=1}^{N} \frac{(u_i(t) - u_{i-1}(t))^2}{2R_i} \Rightarrow \dot{W}(U(t)) = \sum_{i=1}^{N-1} \dot{u}_i(t) \left(\frac{u_i(t) - u_{i-1}(t)}{R_i} + \frac{u_i(t) - u_{i+1}(t)}{R_{i+1}} \right)$$

($W(U(t))$ represents the power dissipated on resistors)

The Distributed Model 2/2

State-Space model: $U = [u_0 \ u_1 \ \ldots \ u_N]^\top$

$$F = \begin{bmatrix} 0 & 0 & 0\ldots & 0 \\ \frac{1}{C_1R_1} & -\frac{R_1+R_2}{C_1R_1R_2} & \frac{1}{C_1R_2}\ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \ldots\frac{1}{C_{N-1}R_{N-1}} & -\frac{R_{N-1}+R_N}{C_{N-1}R_{N-1}R_N} & \frac{1}{C_{N-1}R_N} \\ 0 & 0 & 0\ldots & 0 \end{bmatrix} \rightsquigarrow \dot{U}(t) = FU(t)$$

- Continuous time, **constrained** version:

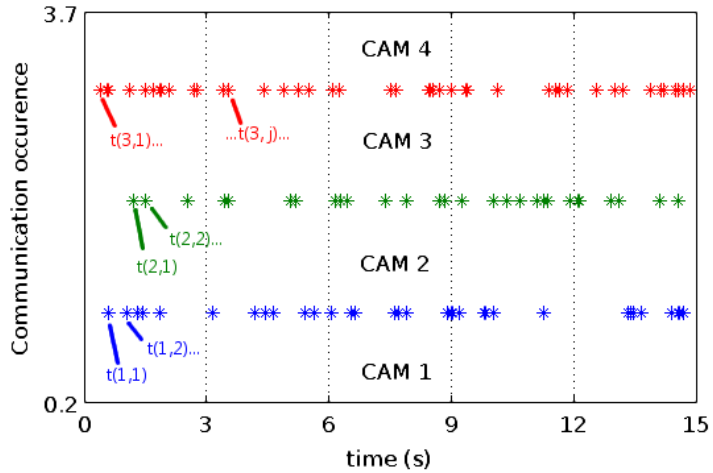
$$\dot{u}_j(t) \equiv 0 \rightsquigarrow \dot{W}(U(t)) = \sum_{i=1}^{j-1}[\ldots] + \sum_{i=j+1}^{N-1}[\ldots], \quad t \geq \bar{t}$$

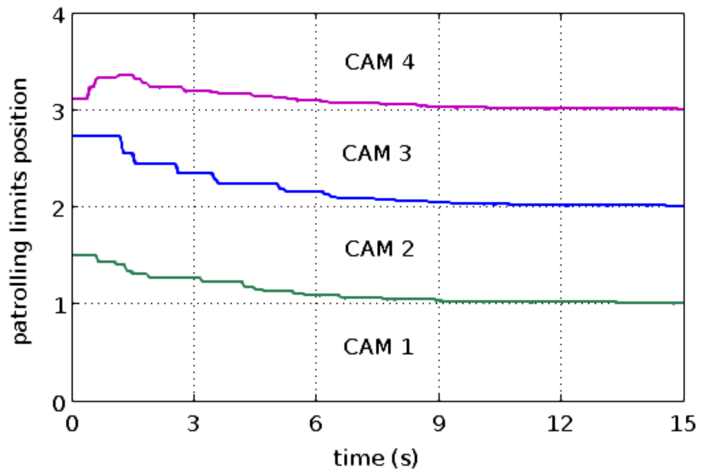
- **Discrete** time, **constrained** version: discretization method with $\frac{\Delta}{C_i}$ small
- **Asynchronous** communication scheme:

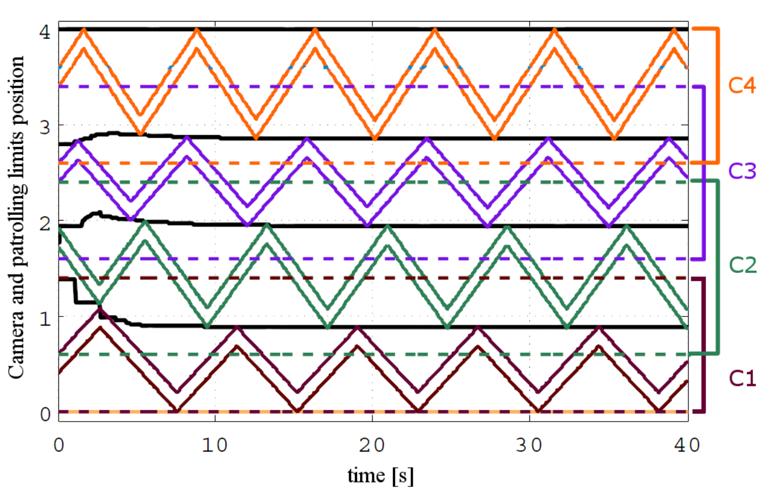
$$a_i^+(t_{i,j}) = k_{i-1}a_{i-1}(t_{i,j}) + k_i a_i(t_{i,j}) + k_{i+1}a_{i+1}(t_{i,j}),$$

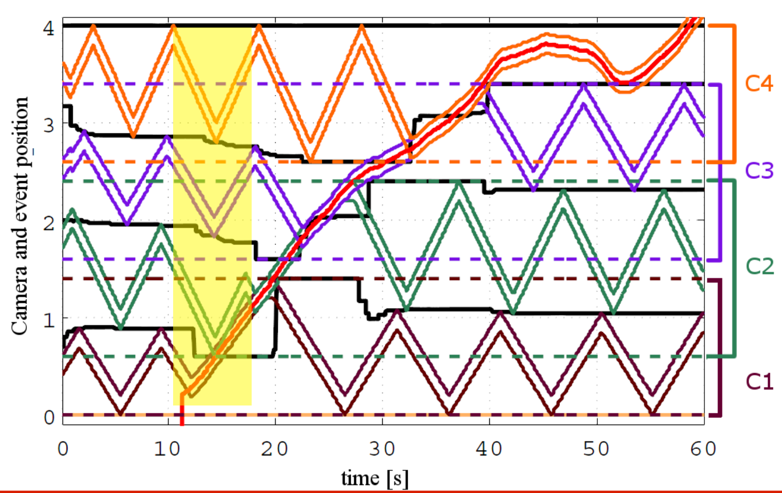
$$k_{i-1} = \frac{\Delta}{C_iR_i} \quad k_i = 1 - \frac{\Delta(R_i + R_{i+1})}{C_iR_iR_{i+1}} \quad k_{i+1} = \frac{\Delta}{C_iR_{i+1}}$$

Simulations









M.Baseggio, A.Cenedese, P.Merlo, M.Pozzi, L.Schenato,
Distributed perimeter patrolling and tracking for camera networks, submitted to CDC10

Outline of the Talk

Related Projects

- **FeedNetBack - Feedback design for wireless networked systems.**

  EU STREP Project ICT Call 2 FP7-ICT-2007-2, 2008-11

- **SIMEA: Integrated/Distributed System for Energetic and Environmental Monitoring.**

  Funded by Regione Veneto, 2010-2012

- **Localization and tracking sensor network system.**

  Funded by Regione Veneto, 2010

- **WISEWAI - Wireless Sensor networks for city-Wide Ambient Intelligence.**

  Funded by CaRiPaRo Foundation, 2007-2010

- **Enhancement on Magnetic Diagnostics. UKAEA-JET (UK).**

  Funded by EU EFDA-JET, 2004-07

- **Smart Environments: event interpretation, sensor reconfiguration, multimodal interfaces**

  Funded by Italian Ministry of University and Research , 2007-09