COOPERATIVE CONTROL AND OPTIMIZATION IN AN UNCERTAIN, ASYNCHRONOUS WIRELESS, NETWORKED WORLD

C. G. Cassandras

Division of Systems Engineering

Center for Information and Systems Engineering

Boston University

ACKNOWLEDGEMENTS:

PhD Students: Wei Li, Ning Xu, Minyi Zhong, Jianfeng Mao, Chen Yao, Yanfeng Geng

Sponsors: NSF, AFOSR, ARO, ONR, DOE, Honeywell

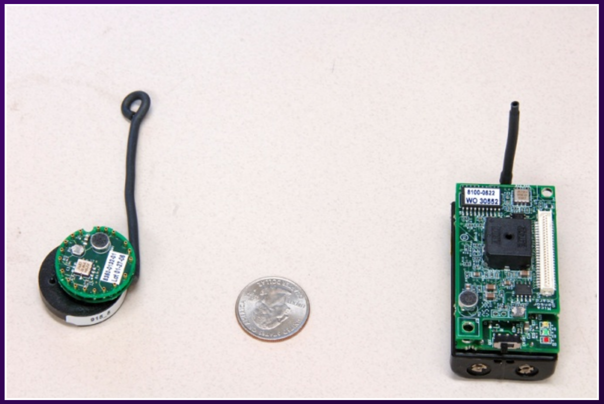
OUTLINE

- Sensor Networks (*"Earth's skin…"*)
- Sensor Networks as Control Systems:
  - Three functions: Coverage – Detection – Data Collection
- Coverage:
  - Distributed Cooperative Optimization
  - Emphasis on Event-Driven control
- Coverage + Data Collection
- Data Collection:
  - Stochastic Multi-Traveling-Salesman Problem with Time-Varying City Rewards
  - Cooperative Receding Horizon (CRH) Control
- DEMOS: Applets and Movies

WHAT'S A SENSOR NETWORK ?

A NETWORK consisting of devices (sensors) that:

- ... communicate wirelessly
- ... are battery-powered
- ... may have different characteristics
- ... have limited processing capabilities
- ... have limited life
- ... often operate in noisy/adversarial environments
- ... monitor/control physical processes

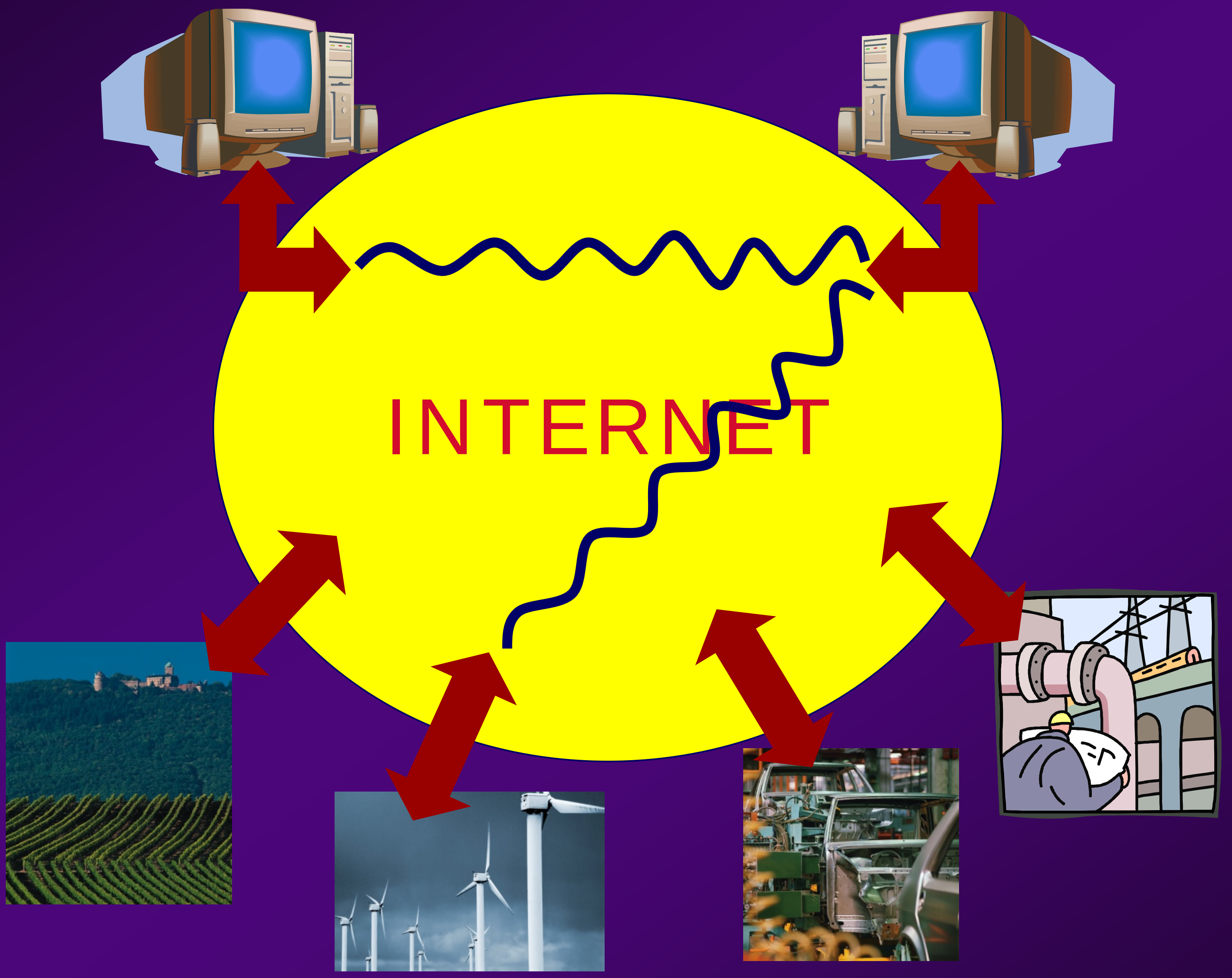


WHY ARE SENSOR NETWORKS EXCITING?

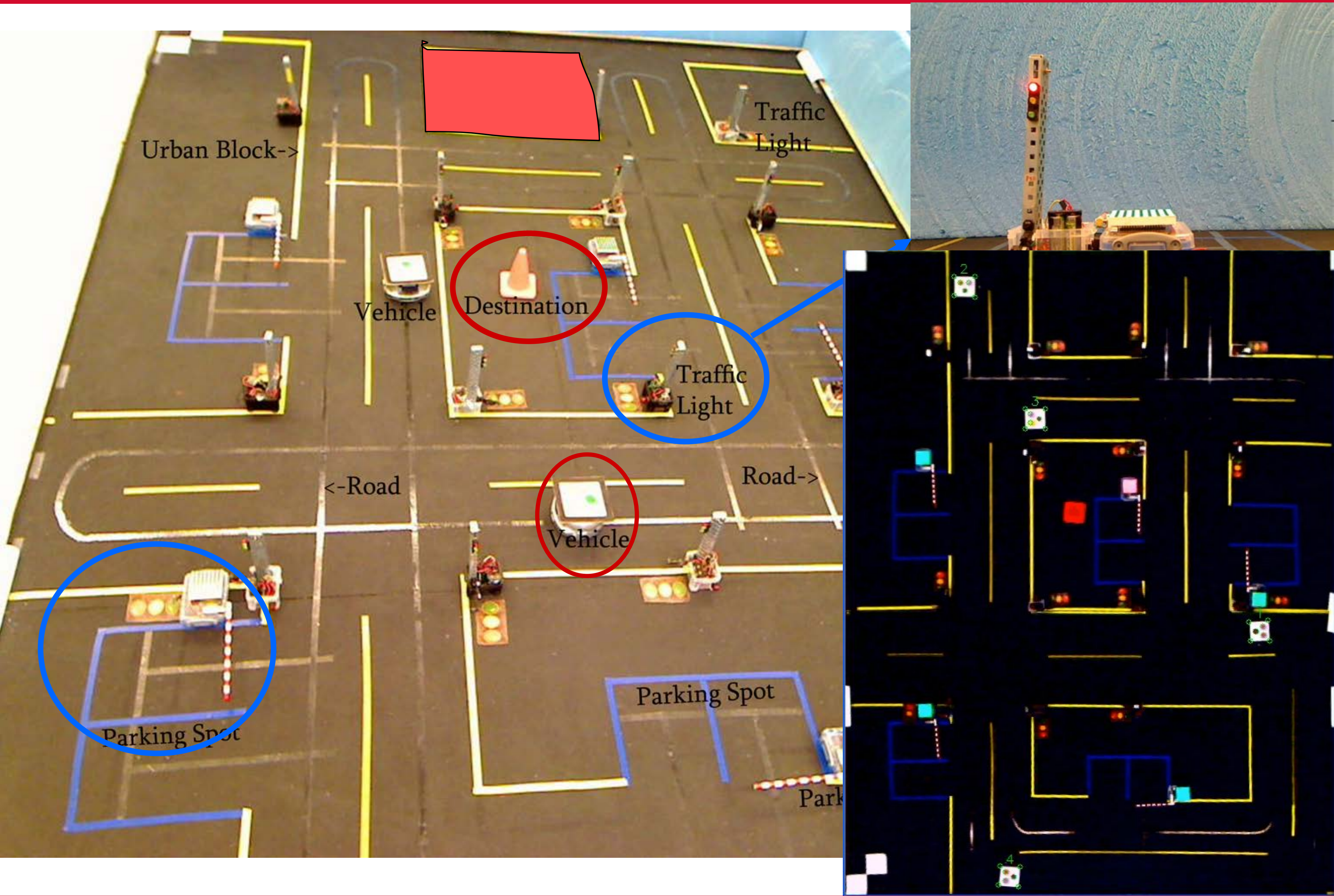
- They interact with the *physical* world
- They promise *fascinating applications*:
  - Smart Buildings (locate persons/objects, find closest resource, adjust environment, detect emergency conditions)
  - Smart Cities (smart parking, location-based services, traffic control)
  - Health monitoring
  - Security and military applications
  - Environmental monitoring
  - Inventory monitoring/replenishment (smart shelves)
  - Equipment condition monitoring, active maintenance (smart appliances)
  - Asset tracking and management (warehouses, ports)
- They realize a convergence of the "3 Cs":

  Communication + Computing + Control

CYBER-PHYSICAL SYSTEMS



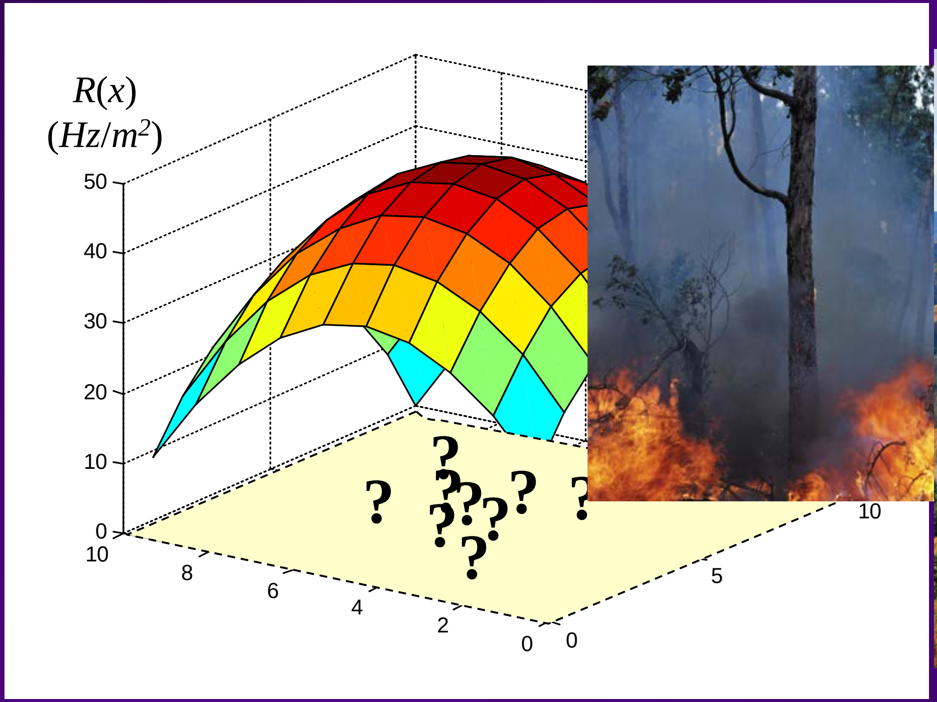
INTELLIGENT PARKING TEST BED

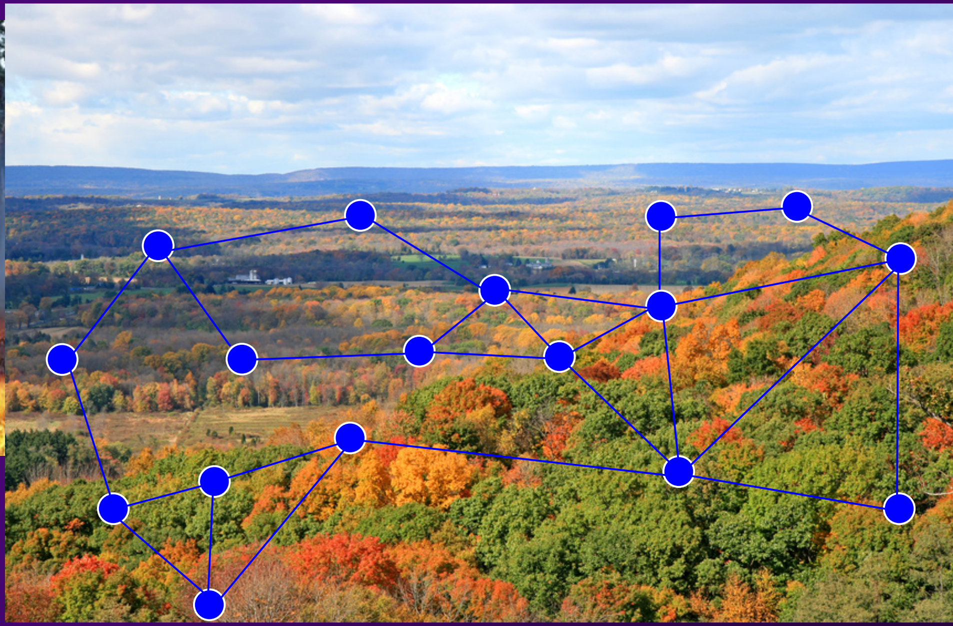


MOTIVATIONAL PROBLEM: *COVERAGE CONTROL*

Deploy sensors to maximize "event" detection probability

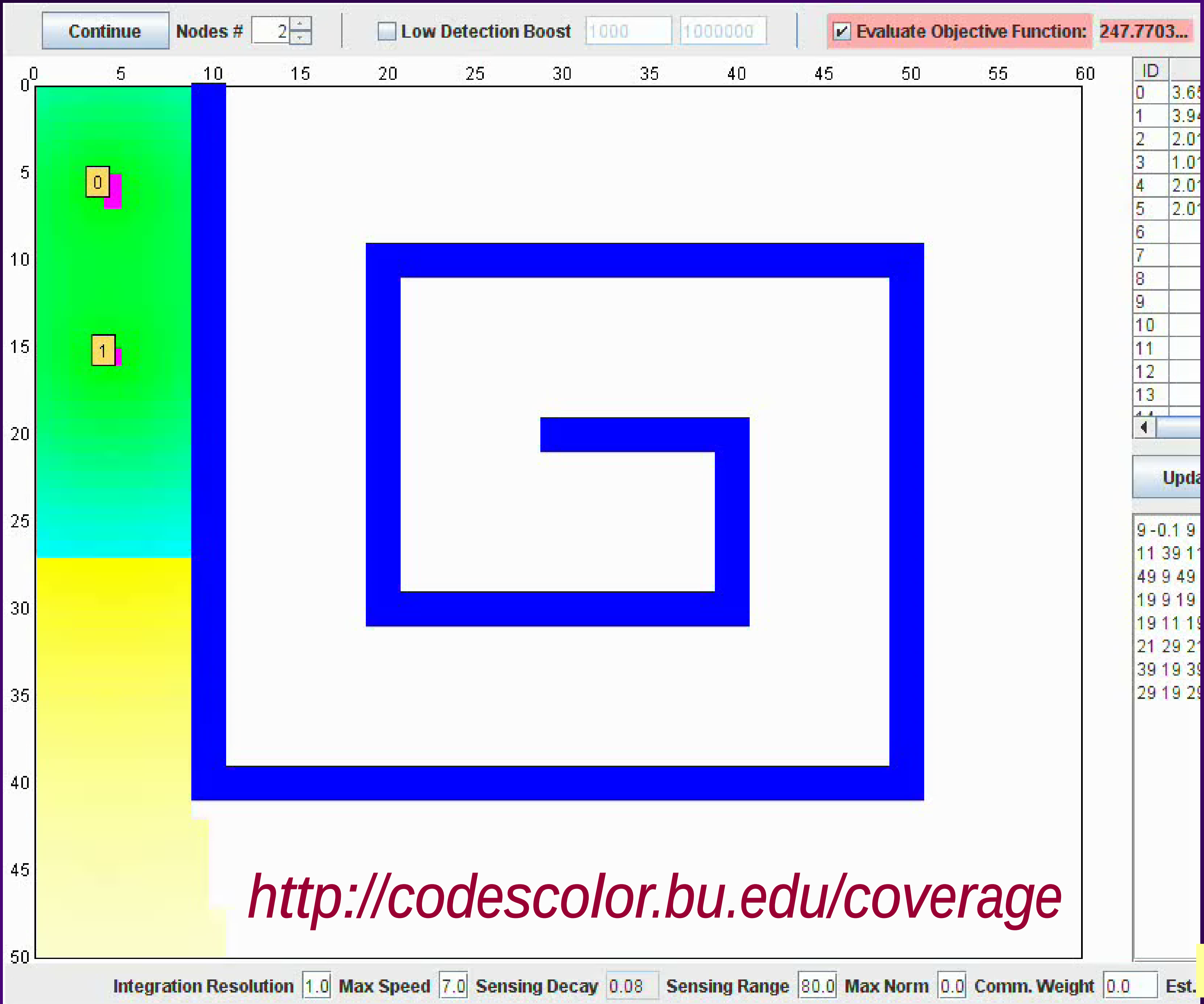
– unknown event locations
– event sources may be mobile
– sensors may be mobile

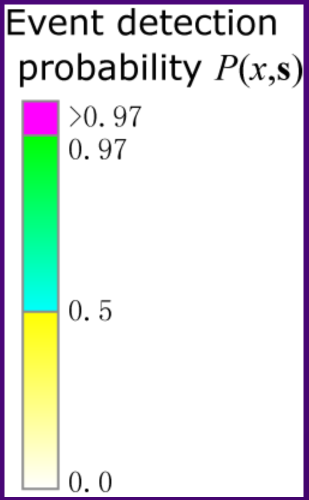




Perceived event density (data sources) over given region (mission space)

OPTIMAL COVERAGE WITH OBSTACLES

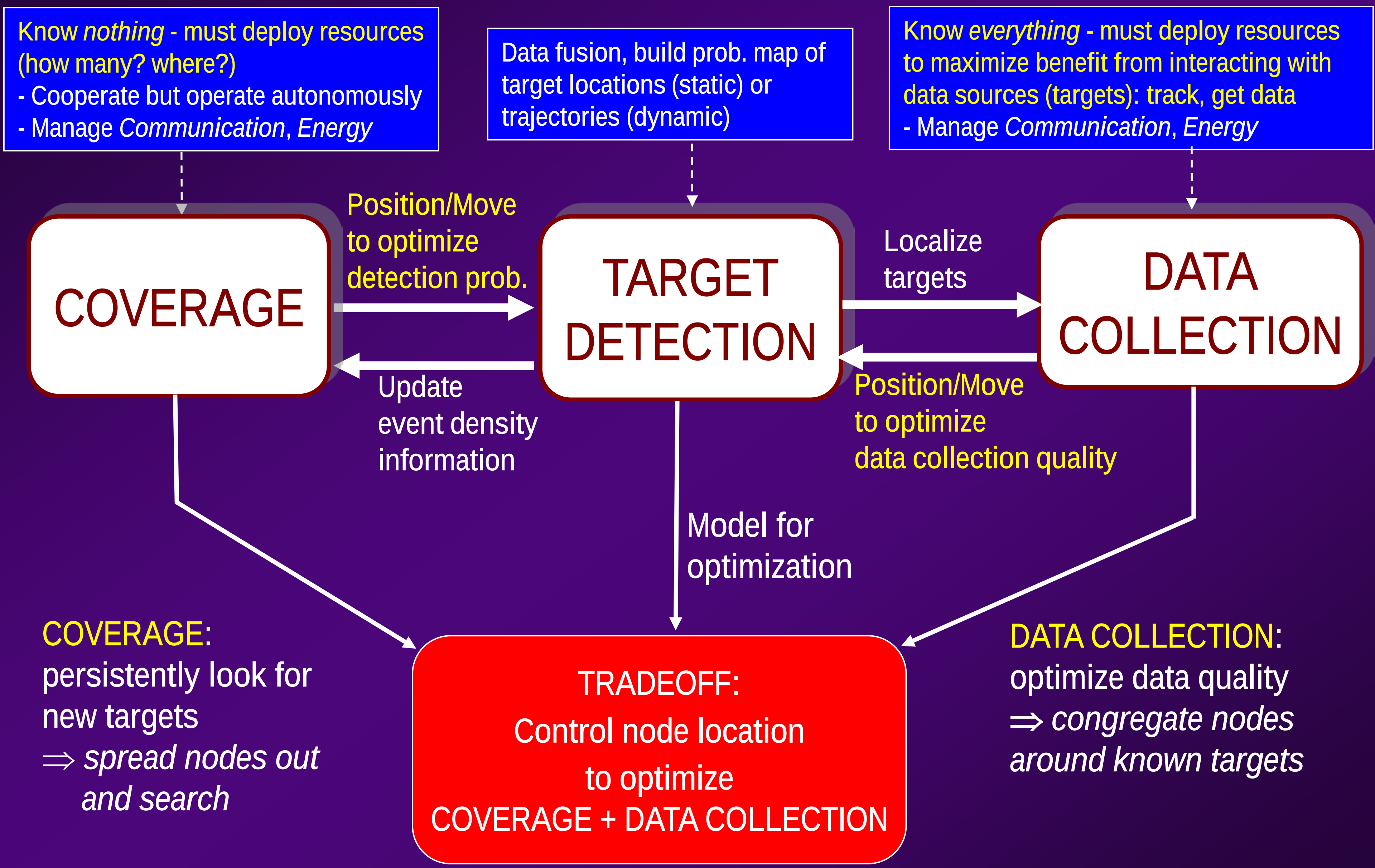




http://codescolor.bu.edu/coverage

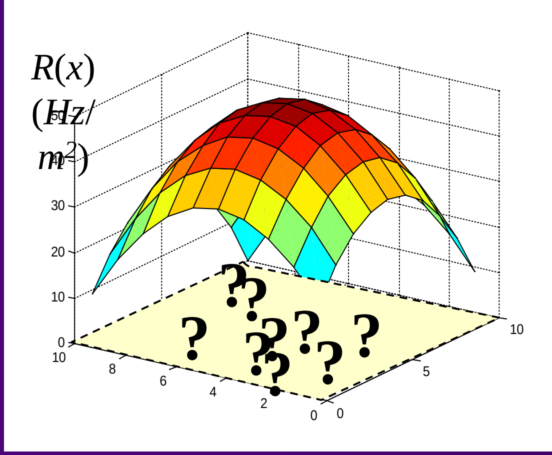
Zhong and Cassandras, 2008

SENSOR NETWORK AS A CONTROL SYSTEM



THE COVERAGE PROBLEM

COVERAGE: PROBLEM FORMULATION



- N mobile sensors, each located at $s_i \in \mathbf{R}^2$
- Data source at x emits signal with energy E
- Signal observed by sensor node i (at s_i)
- SENSING MODEL:

$$p_i(x, s_i) \equiv P[\text{Detected by } i \mid A(x), s_i]$$

($A(x)$ = data source emits at x)

- Sensing attenuation:
  $p_i(x, s_i)$ monotonically decreasing in $d_i(x) \equiv \|x - s_i\|$

COVERAGE: PROBLEM FORMULATION

- Joint detection prob. assuming sensor independence ($\mathbf{s} = [s_1,\ldots,s_N]$: node locations)

Event sensing probability

$$P(x,\mathbf{s}) = 1 - \prod_{i=1}^{N}\left[1 - p_i(x,s_i)\right]$$

- OBJECTIVE: Determine locations $\mathbf{s} = [s_1,\ldots,s_N]$ to maximize total *Detection Probability*:

$$\max_{\mathbf{s}} \int_{\Omega} R(x)P(x,\mathbf{s})dx$$

Perceived event density

DISTRIBUTED COOPERATIVE SCHEME

- Set

$$H(s_1,\ldots,s_N)=\int_{\Omega}R(x)\left\{1-\prod_{i=1}^{N}\left[1-p_i(x)\right]\right\}dx$$

- Maximize $H(s_1,\ldots,s_N)$ by forcing nodes to move using gradient information:

$$\frac{\partial H}{\partial s_k}=\int_{\Omega}R(x)\prod_{i=1,i\neq k}^{N}\left[1-p_i(x)\right]\frac{\partial p_k(x)}{\partial d_k(x)}\frac{s_k-x}{d_k(x)}dx$$

$$s_i^{k+1}=s_i^{k}+\beta_k\frac{\partial H}{\partial s_i^{k}}$$

Desired displacement $= V\cdot\Delta t$

DISTRIBUTED COOPERATIVE SCHEME *CONTINUED*

$$\frac{\partial H}{\partial s_k} = \int_{\Omega} R(x) \prod_{i=1, i\neq k}^{N} \left[1 - p_i(x)\right] \frac{\partial p_k(x)}{\partial d_k(x)} \frac{s_k - x}{d_k(x)} dx$$

... has to be autonomously evaluated by each node so as to determine how to move to next position:

$$s_i^{k+1} = s_i^k + \beta_k \frac{\partial H}{\partial s_i^k}$$

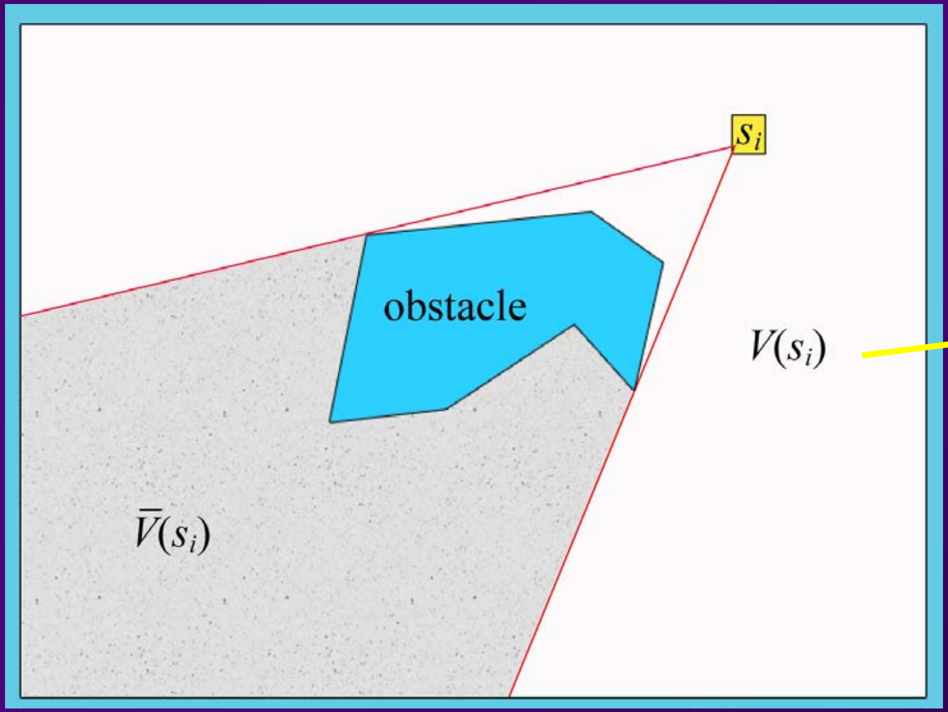
- Use truncated $p_i(x) \Rightarrow \Omega$ replaced by node neighborhood
- Discretize $p_i(x)$ using a local grid

Cassandras and Li, 2005

EXTENSION 1: POLYGONAL OBSTACLES…

- Constrain the navigation of mobile nodes

- Interfere with sensing: $$\hat{p}_i(x, s_i) = \begin{cases} p_i(x, s_i) & \text{if } x \text{ is visible from } s_i \\ 0 & \text{otherwise} \end{cases}$$



Visibility Region

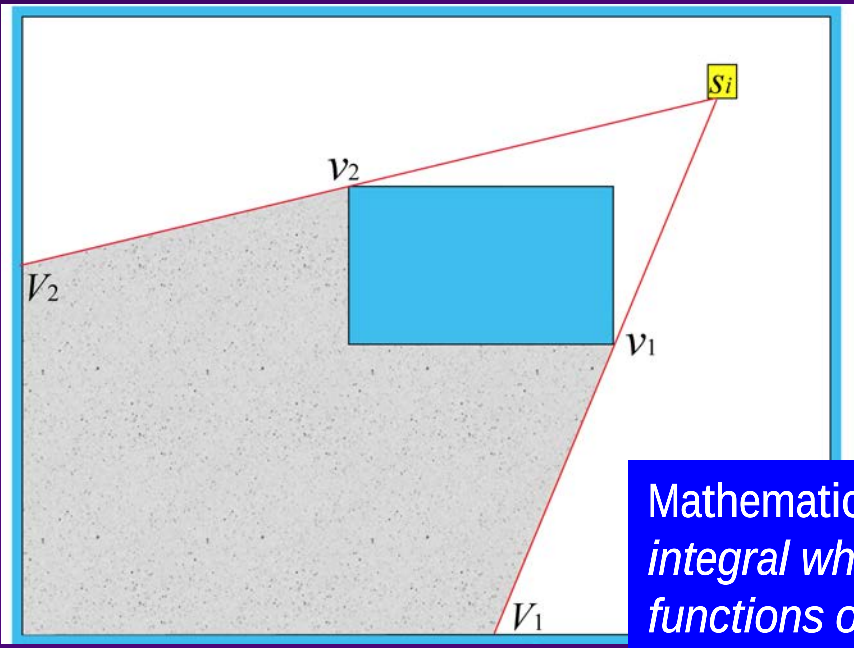
GRADIENT CALCULATION WITH OBSTACLES

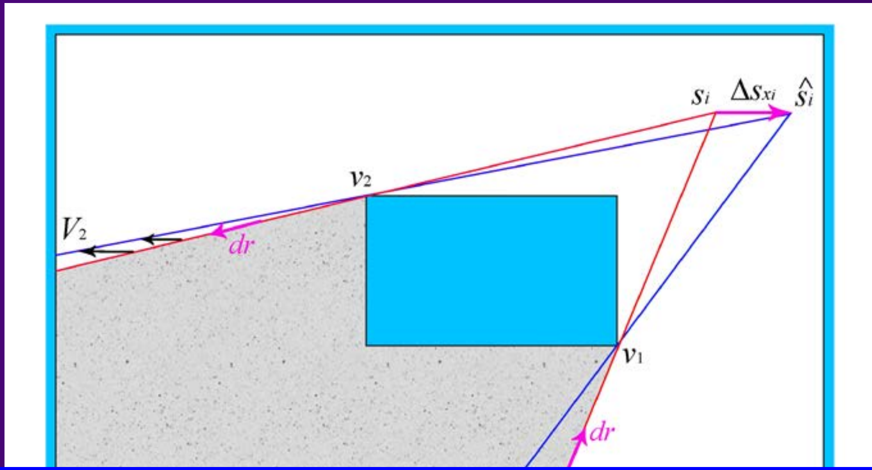
$$\frac{\partial H}{\partial s_i} = \int_{\boxed{V(s_i)}} R(x) \prod_{k=1,k\neq i}^{N} [1-\hat{p}_k(x,s_k)] \frac{\partial \hat{p}_i(x,s_i)}{\partial d_i(x)} \frac{s_i - x}{d_i(x)} dx + \sum_{j=1}^{Q(s_i)} A_j$$

$$\hat{p}_i = \begin{cases} p_i & \text{visible} \\ 0 & \text{invisible} \end{cases}$$

$Q(s_i)$: # of occluding corner points

New term captures change in visibility region of s_i

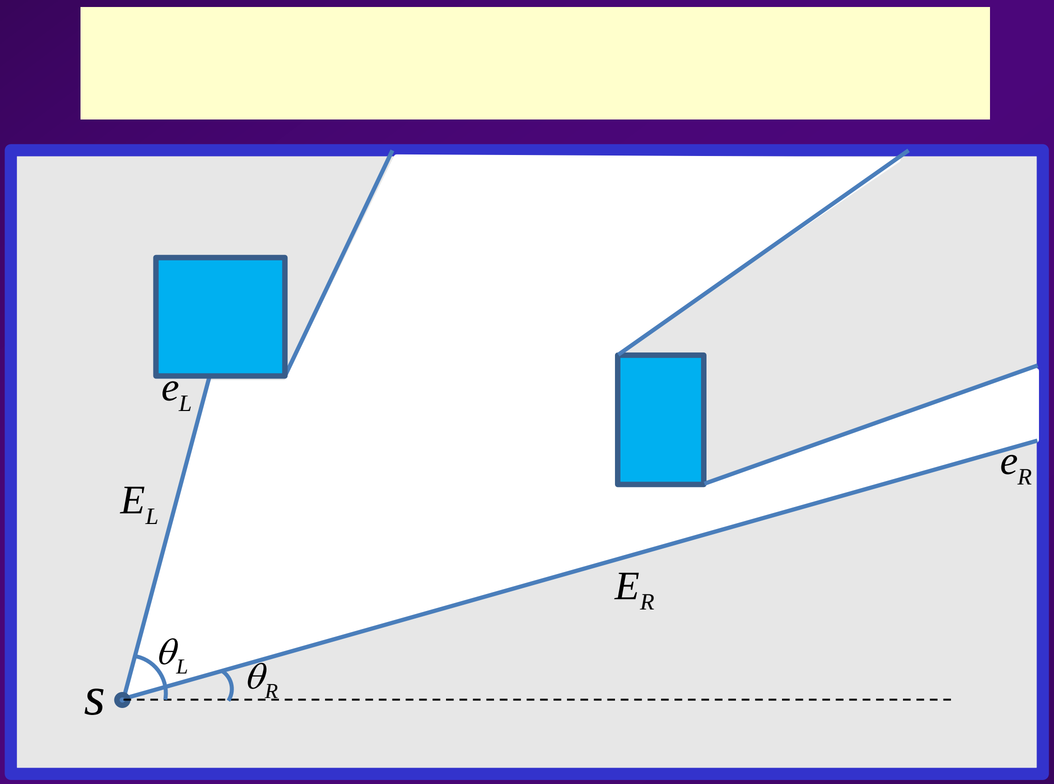




Mathematically: *use extension of Leibnitz rule for differentiating integral where both integrand and integration domain are functions of the control variable*

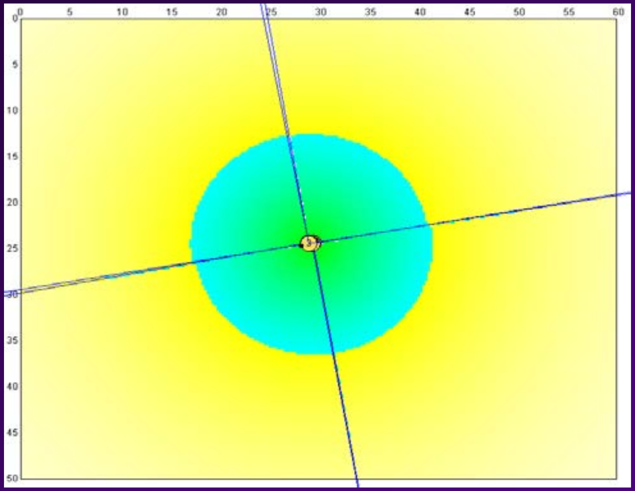
EXTENSION 2: LIMITED FIELD OF VIEW…

- Sensors (e.g., cameras) may have limited field of view (FOV)
- Modeled as a sensing cone with a fixed aperture
- New control variable at each node: FOV direction θ_i

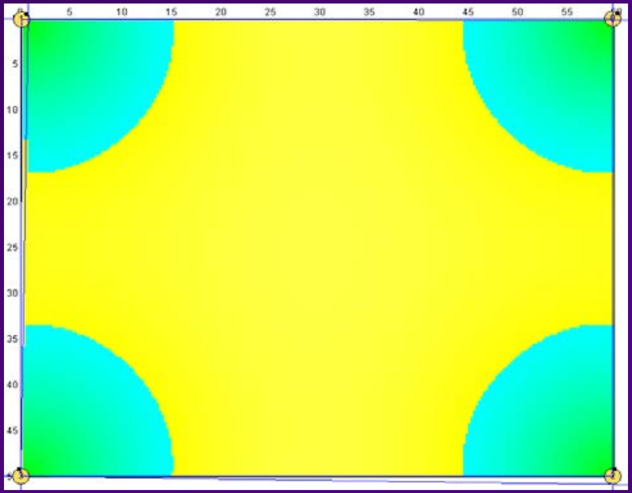


- Edges of sensing cone introduce discontinuities similar to those introduced by obstacles $\Rightarrow$ similar gradient evaluation

LIMITED FIELD OF VIEW - EXAMPLES

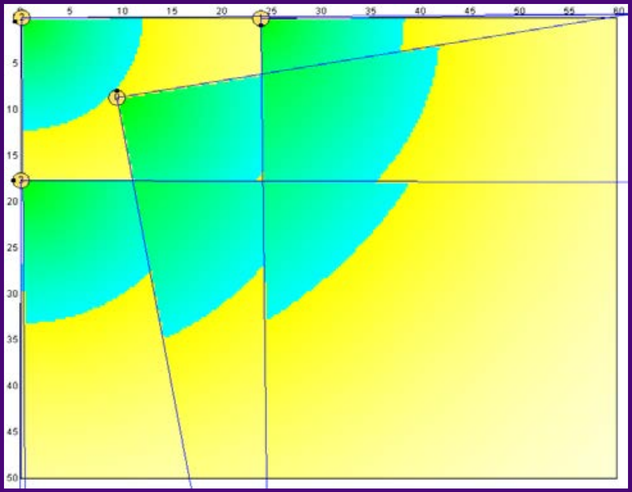


FOV = 90º, start from center, (991)

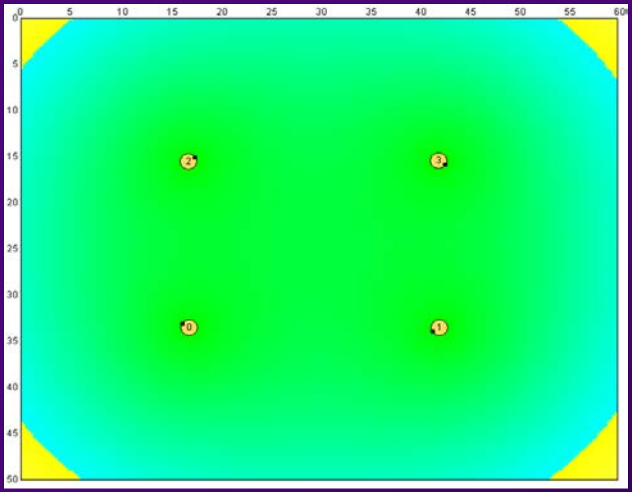


FOV = 90º, start from 4 corners, (1404)

PERFORMANCE



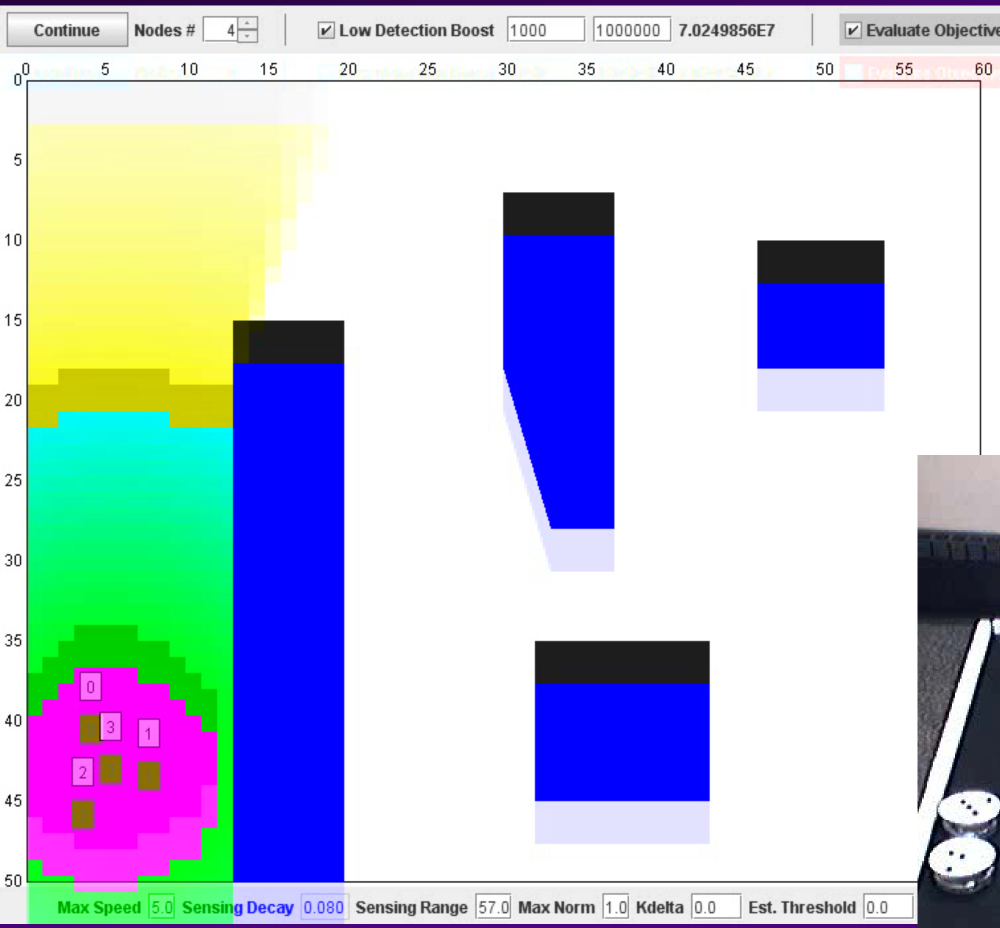
FOV = 90º, start from 1 corner, (1249)

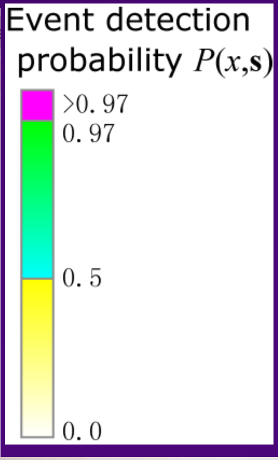


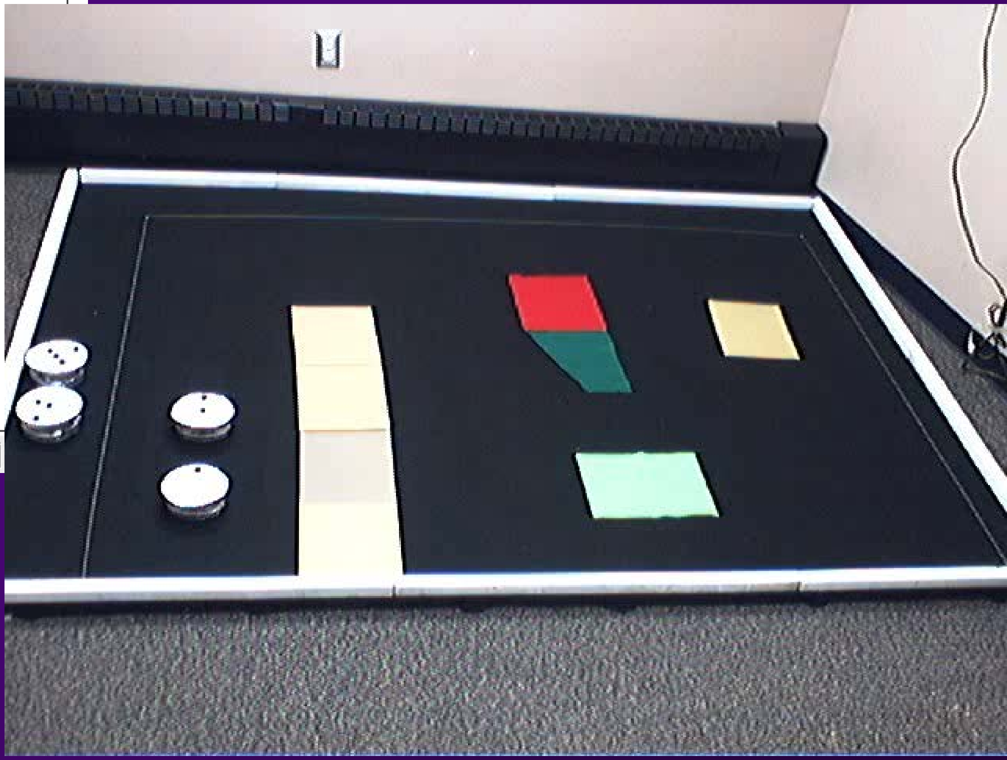
FOV = 360º degree, (2290)

Final configuration strongly depends on initial condition

DEMO: OPTIMAL DISTRIBUTED DEPLOYMENT WITH OBSTACLES – *SIMULATED AND REAL*





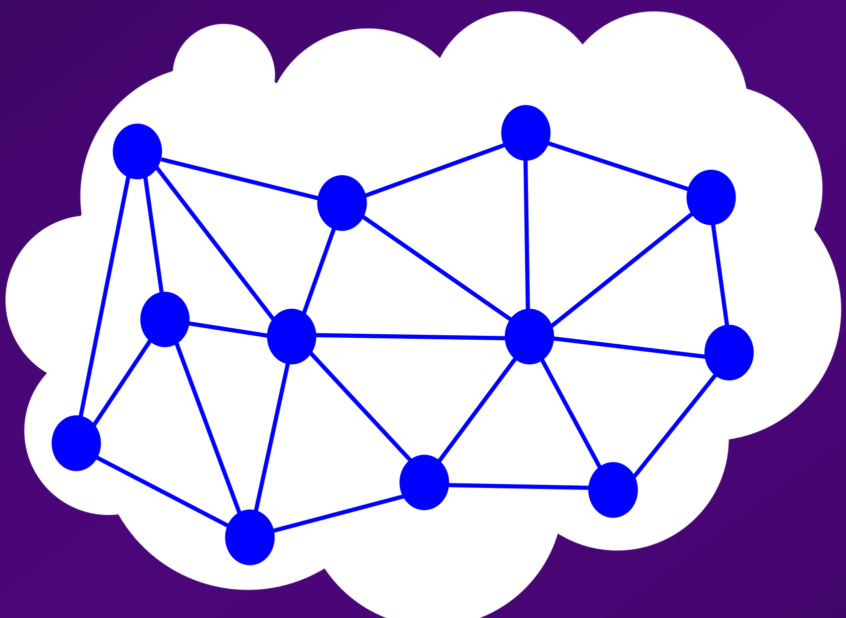


THE BIGGER PICTURE: DISTRIBUTED OPTIMIZATION

DISTRIBUTED COOPERATIVE OPTIMIZATION

N system components (processors, agents, vehicles, nodes), one common objective:

$$\min_{s_1,\ldots,s_N} H(s_1,\ldots,s_N)$$
$$s.t. \text{ constraints on each } s_i$$

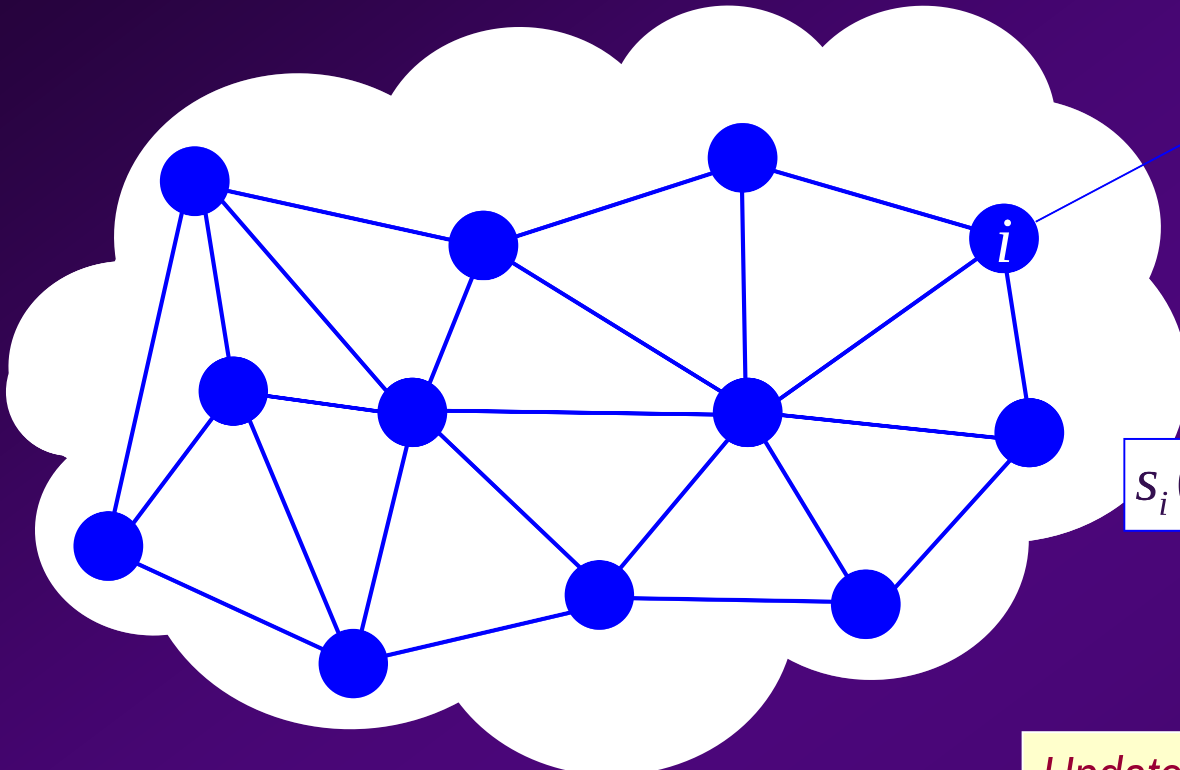


$$\min_{s_1} H(s_1,\ldots,s_N)$$
$$s.t. \text{ constraints on } s_1$$

$\vdots$

$$\min_{s_N} H(s_1,\ldots,s_N)$$
$$s.t. \text{ constraints on } s_N$$

DISTRIBUTED COOPERATIVE OPTIMIZATION



Controllable *state*

$s_i,\ i = 1,\ldots,n_i$

$$s_i(k+1) = s_i(k) + \alpha_i d_i(\mathbf{s}(k))$$

Step Size

Update Direction, usually

$$d_i(\mathbf{s}(k)) = -\nabla_i H(\mathbf{s}(k))$$

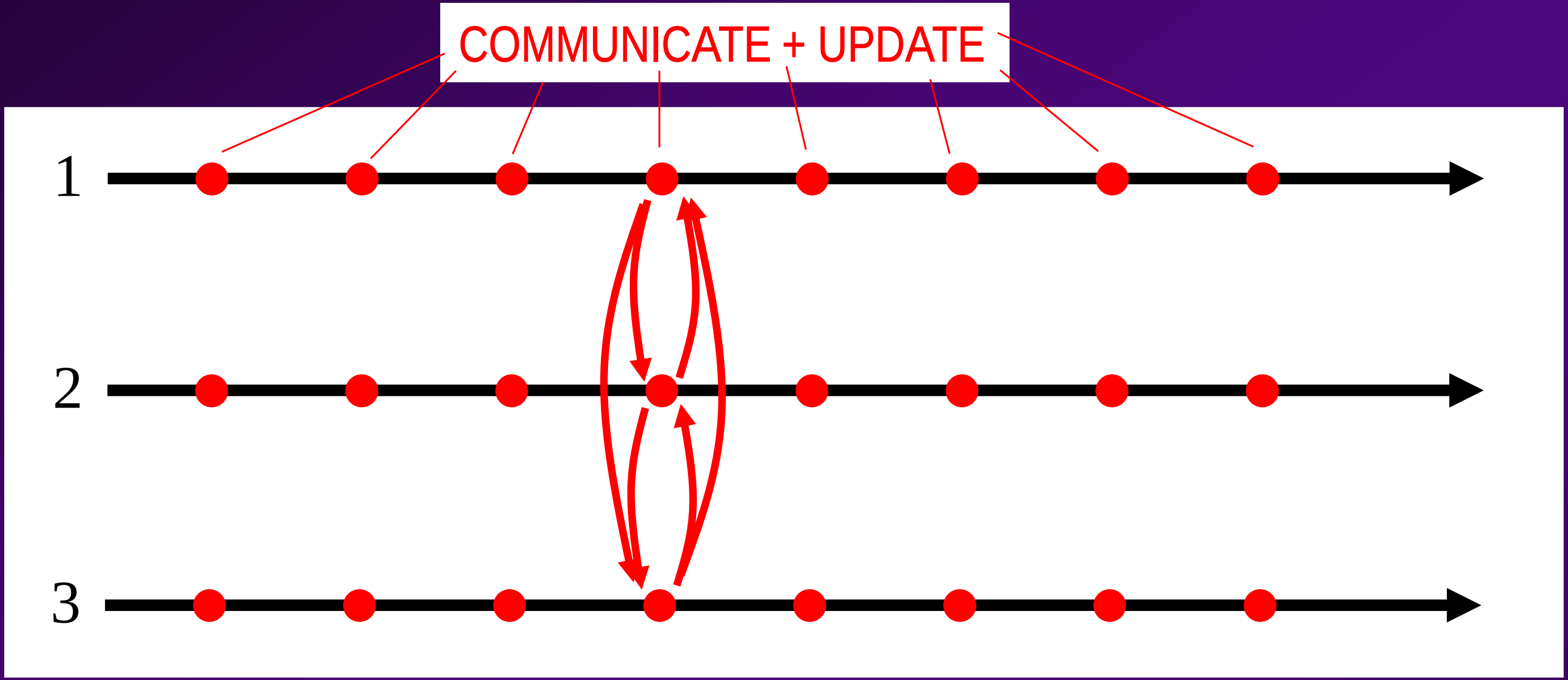
i requires knowledge of all $s_1,\ldots,s_N$

Inter-node communication

$$\min_{s_i} H(s_1,\ldots,s_N)$$

$s.t.$ constraints on s_i

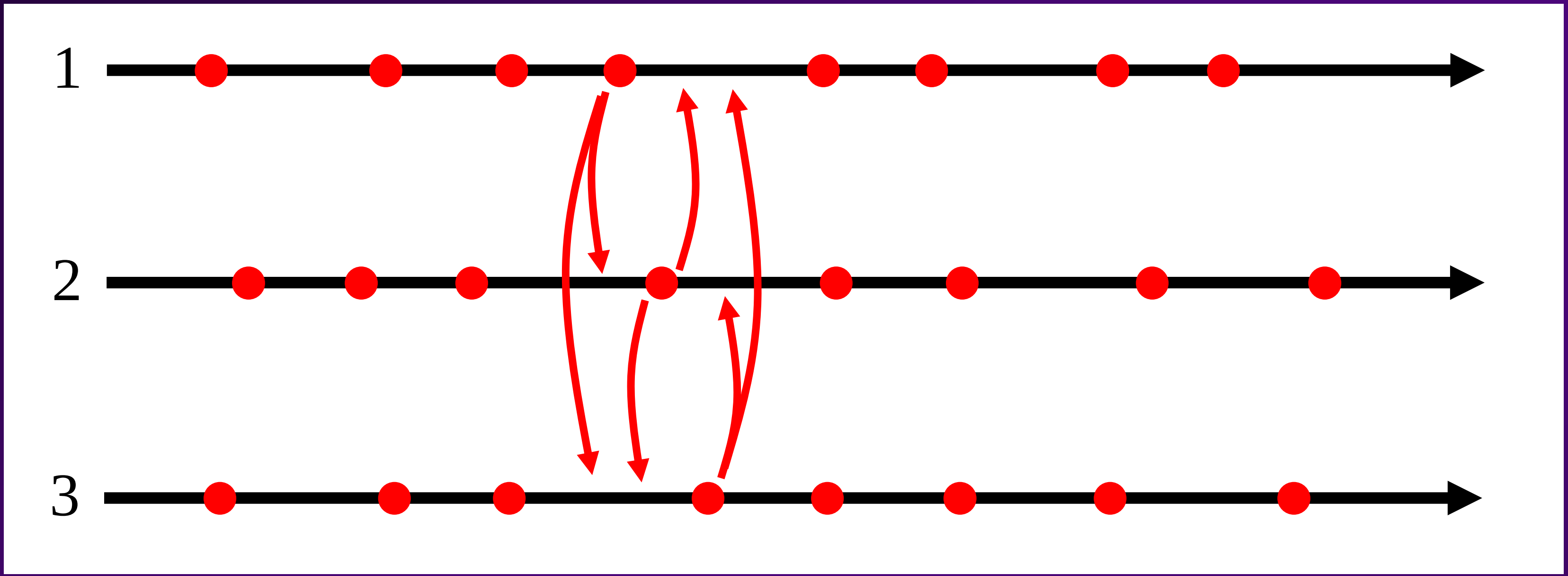
SYNCHRONIZED (TIME-DRIVEN) COOPERATION



Drawbacks:

- Excessive communication (critical in wireless settings!)
- Faster nodes have to wait for slower ones
- Clock synchronization infeasible
- Bandwidth limitations
- Security risks

ASYNCHRONOUS COOPERATION



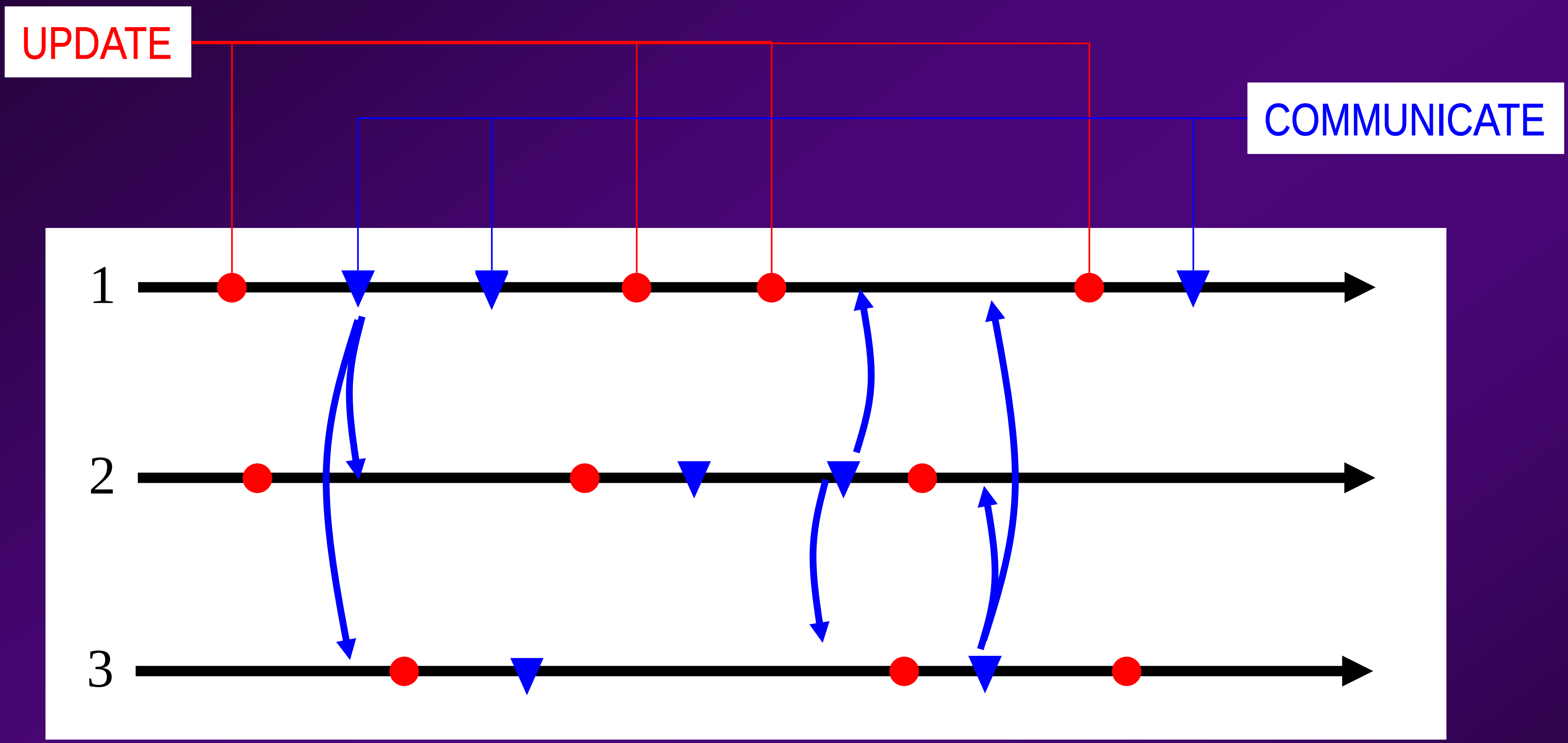
- Nodes not synchronized, delayed information used

Update frequency for each node is bounded
+
technical conditions

$\Rightarrow$ $s_i(k+1) = s_i(k) + \alpha_i d_i(\mathbf{s}(k))$ converges

Bertsekas and Tsitsiklis, 1997

ASYNCHRONOUS (EVENT-DRIVEN) COOPERATION



- UPDATE at i : locally determined, arbitrary (possibly periodic)
- COMMUNICATE from i : only when absolutely necessary

HOW MUCH COMMUNICATION FOR OPTIMAL COOPERATION ?

WHEN SHOULD A NODE COMMUNICATE?

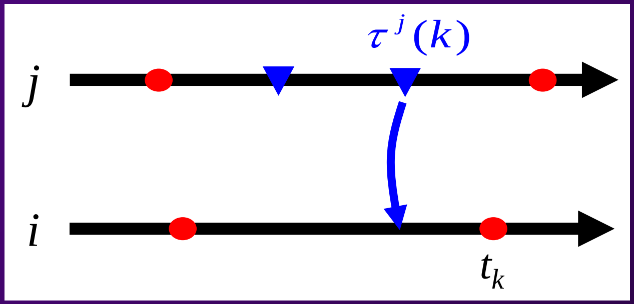
Node state at any time t : $x_i(t)$

Node state at t_k : $s_i(k)$

$\Rightarrow \ s_i(k) = x_i(t_k)$

AT UPDATE TIME t_k , $k \in C^i$: $s_j^i(k)$: node j state estimated by node i

Estimate examples:



➡ $s_j^i(k) = x_j(\tau^j(k))$ Most recent value

➡ $s_j^i(k) = x_j(\tau^j(k)) + \dfrac{t_k - \tau^j(k)}{\Delta_j} \cdot \alpha_i \cdot d_j\left(x_j(\tau^j(k))\right)$ Linear prediction

WHEN SHOULD A NODE COMMUNICATE?

AT ANY TIME t :

- $x_i^j(t)$: node i state estimated by node j
- If node i knows how j estimates its state, then it can evaluate $x_i^j(t)$
- Node i uses
  - its own true state, $x_i(t)$
  - the estimate that j uses, $x_i^j(t)$

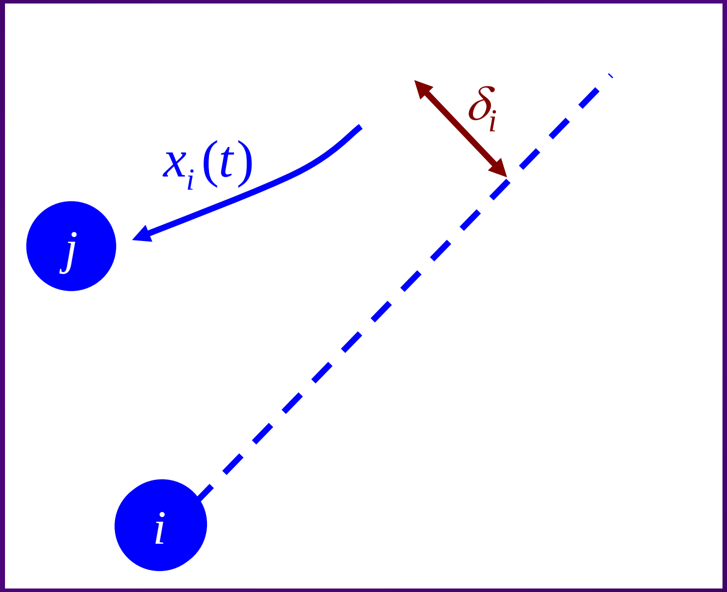
... and evaluates an ERROR FUNCTION $g\left(x_i(t), x_i^j(t)\right)$

Error Function examples: $\left\|x_i(t) - x_i^j(t)\right\|_1, \quad \left\|x_i(t) - x_i^j(t)\right\|_2$

WHEN SHOULD A NODE COMMUNICATE?

Compare ERROR FUNCTION $g\left(x_i(t), x_i^j(t)\right)$ to THRESHOLD δ_i

Node ***i*** communicates its state to node j only when it detects that its *true state* $x_i(t)$ deviates from ***j'*** *estimate of it* $x_i^j(t)$ so that $g\left(x_i(t), x_i^j(t)\right) \geq \delta_i$



$\Rightarrow$ *Event-Driven* Control

CONVERGENCE

Asynchronous distributed state update process at each i:

$$s_i(k+1) = s_i(k) + \alpha \cdot d_i(\mathbf{s}^i(k))$$

$$\delta_i(k) = \begin{cases} K_\delta \left\| d_i(\mathbf{s}^i(k)) \right\| & \text{if } k \in C^i \\ \delta_i(k-1) & \text{otherwise} \end{cases}$$

Estimates of other nodes, evaluated by node i

THEOREM: Under certain conditions, there exist positive constants α and K_δ such that

$$\lim_{k \to \infty} \nabla H(\mathbf{s}(k)) = 0$$

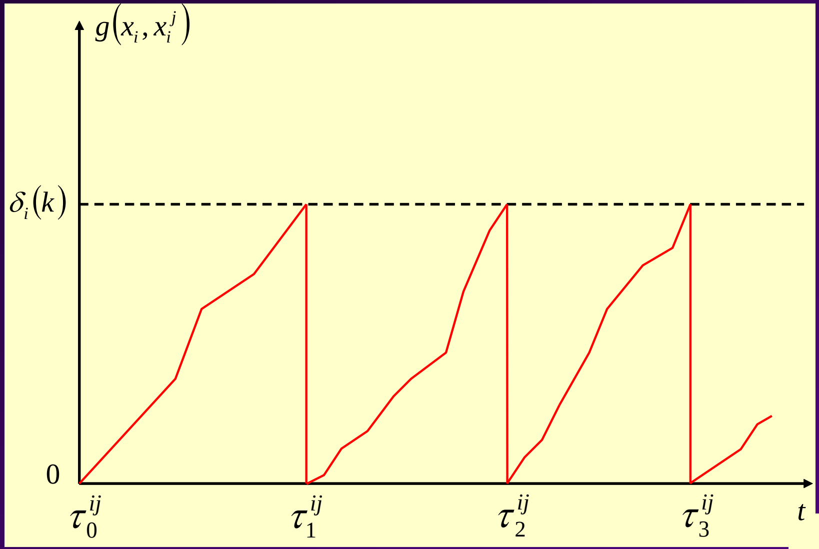
NOTE: Analysis uses framework based on [*Bertsekas and Tsitsiklis, 1997*]

Zhong and Cassandras, 2009

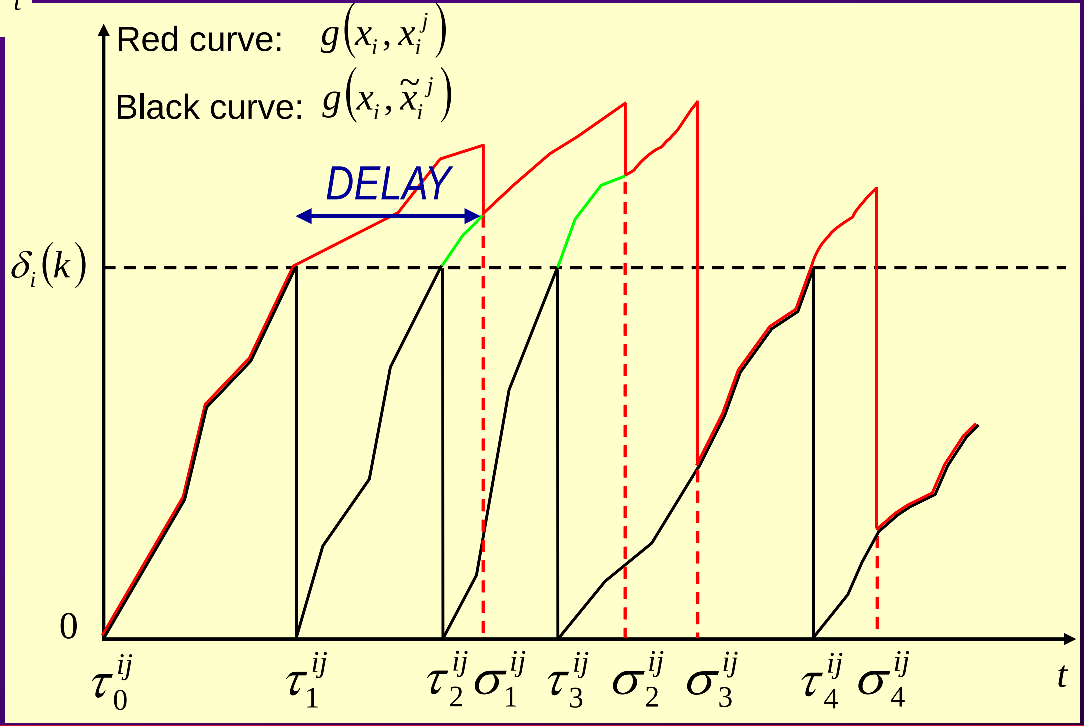
INTERPRETATION:

Event-driven cooperation achievable with minimal communication requirements $\Rightarrow$ *energy savings*

COONVERGENCE WHEN DELAYS ARE PRESENT



Error function trajectory with NO DELAY



COONVERGENCE WHEN DELAYS ARE PRESENT

Add a boundedness assumption:

ASSUMPTION: There exists a non-negative integer D such that if a message is sent before t_{k-D} from node i to node j, it will be received before t_k.

INTERPRETATION: at most $\boldsymbol{D}$ state update events can occur between a node sending a message and all destination nodes receiving this message.

THEOREM: Under certain conditions, there exist positive constants α and K_δ such that

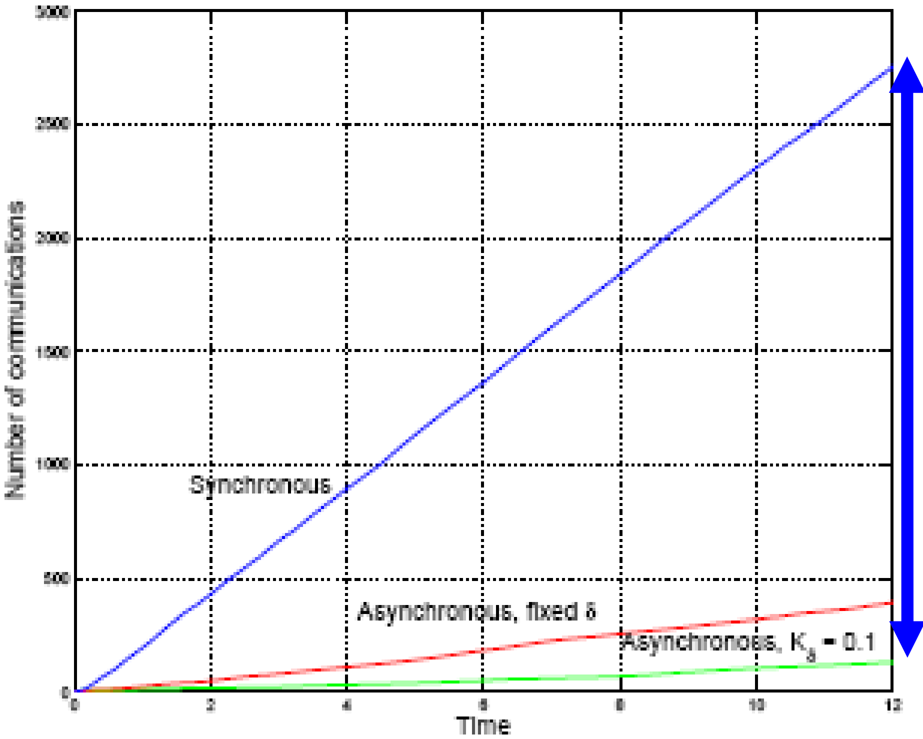
$$\lim_{k\to\infty} \nabla H(\mathbf{s}(k)) = 0$$

NOTE: The requirements on α and K_δ depend on $\boldsymbol{D}$ and they are tighter.

Zhong and Cassandras, 2009

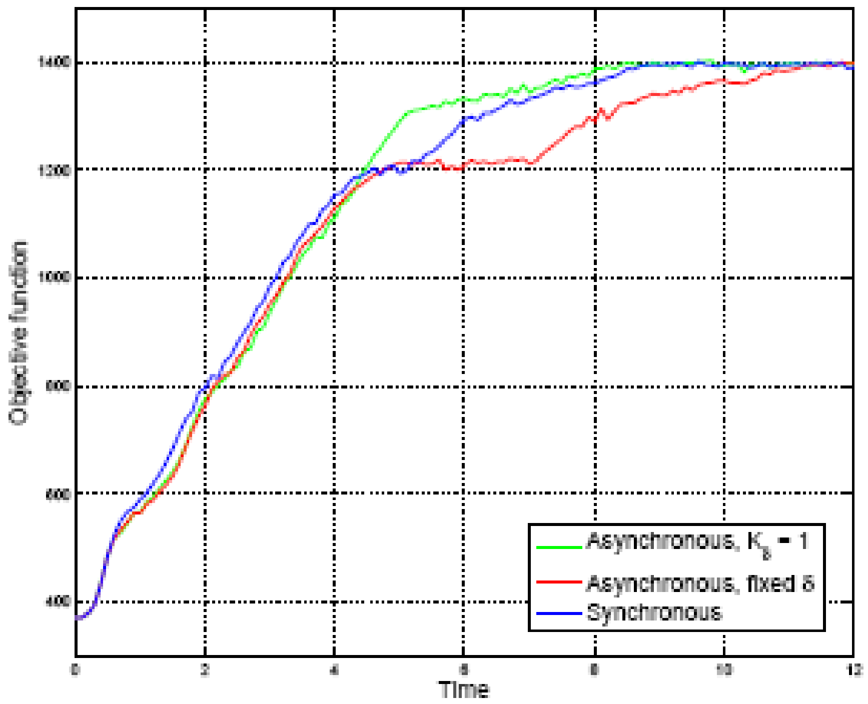
SYNCHRONOUS v ASYNCHRONOUS OPTIMAL COVERAGE PERFORMANCE

Energy savings + Extended lifetime



SYNCHRONOUS v ASYNCHRONOUS:

No. of communication events
for a deployment problem *with obstacles*



SYNCHRONOUS v ASYNCHRONOUS:

Achieving optimality
in a problem *with obstacles*

THE
DATA COLLECTION
PROBLEM

COVERAGE + DATA COLLECTION

Recall tradeoff:

COVERAGE:
persistently look for new targets
$\Rightarrow$ *spread nodes out*

TRADEOFF:
Control node location
to optimize
COVERAGE + DATA COLLECTION

DATA COLLECTION:
optimize data quality
$\Rightarrow$ *congregate nodes around known targets*

MODIFIED DISTRIBUTED OPTIMIZATION OBJECTIVE:

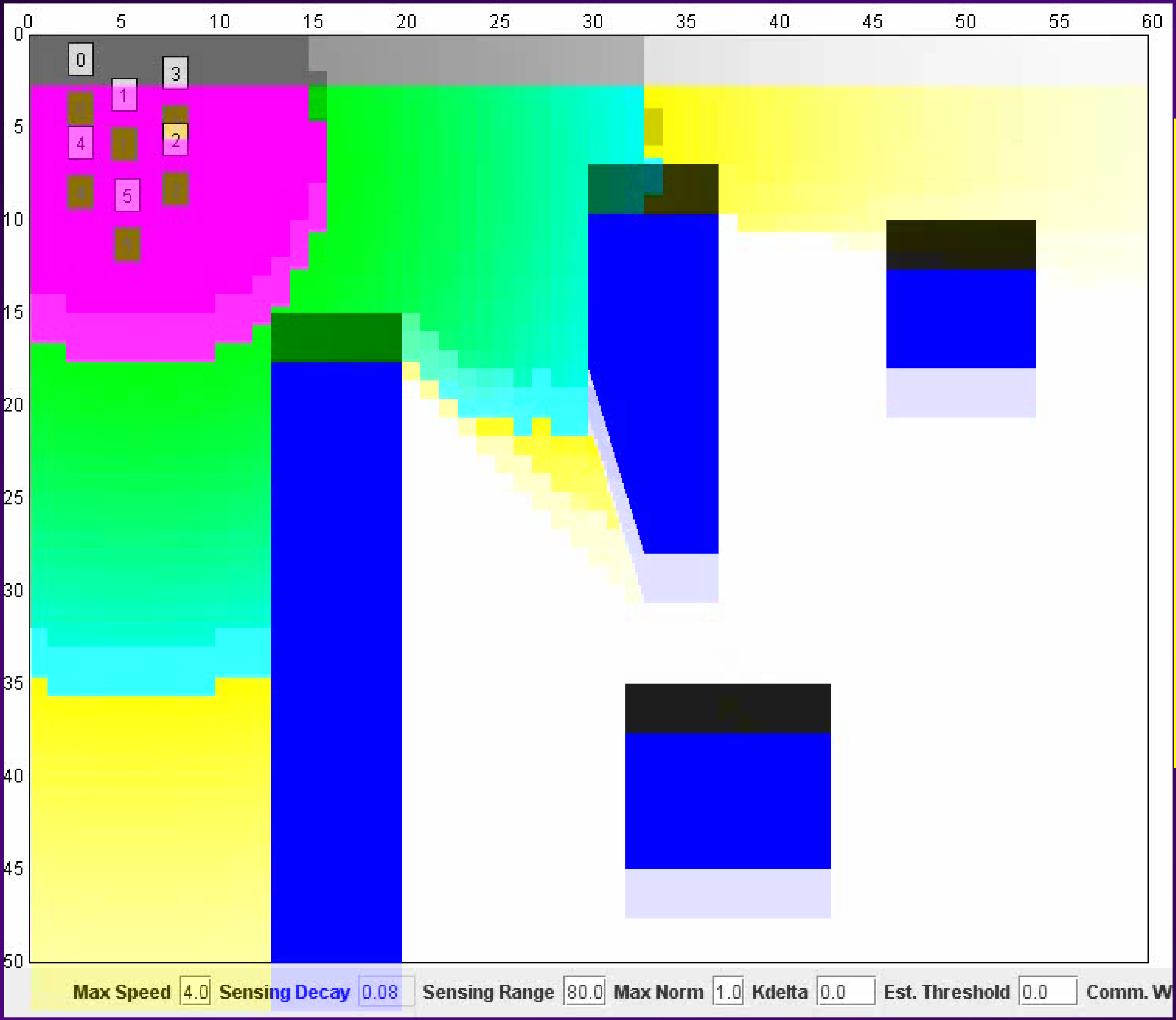
collect info from detected data sources (targets) while maintaining a good coverage to detect future events

$S(u)$: data source value

$\mathcal{D}_t$: set of data sources, estimated based on sensor observations

$F(u,s)$: joint data collection quality at u (e.g., covariance)

DEMO: REACTING TO EVENT DETECTION

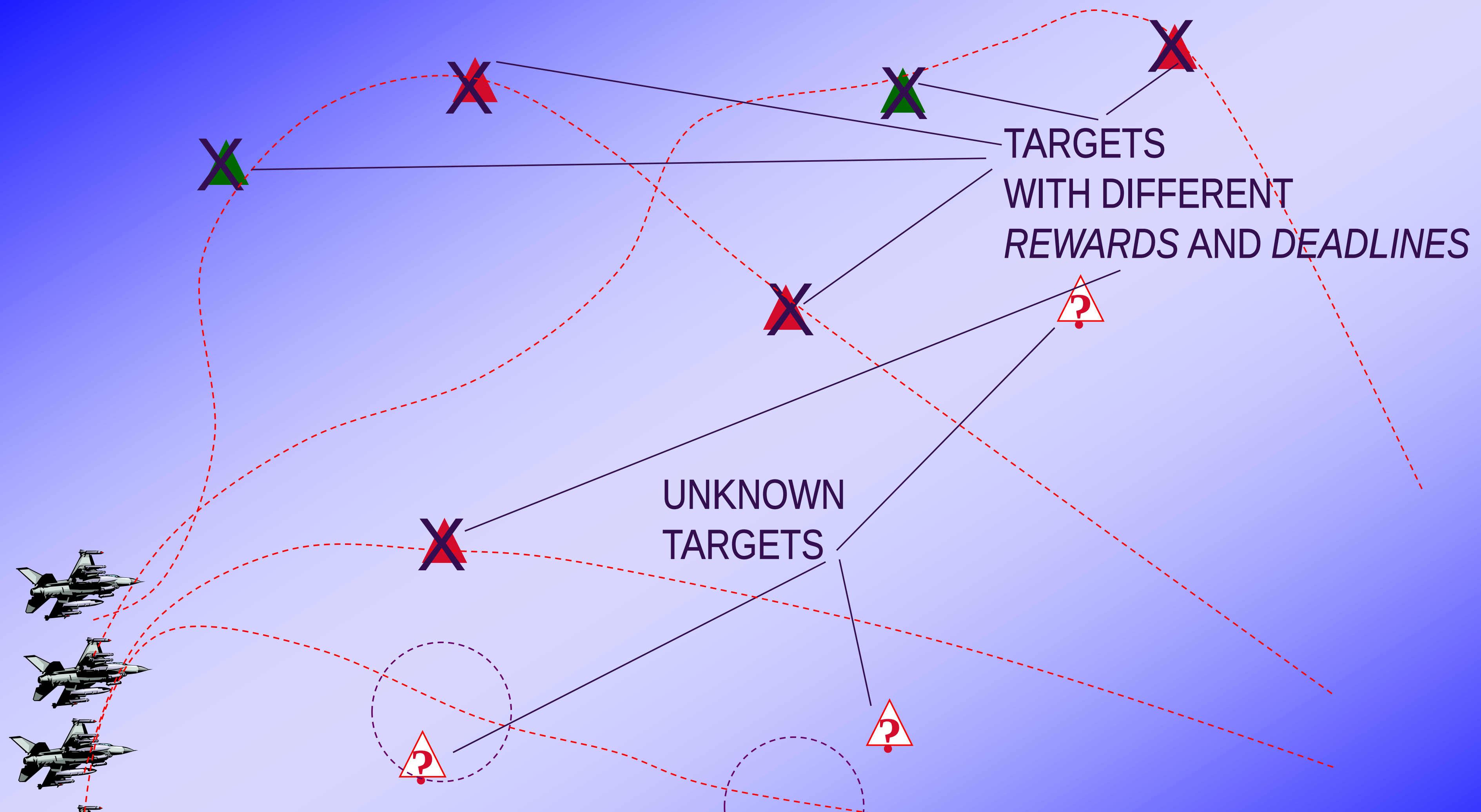


Important to note:

There is no external control causing this behavior. Algorithm includes tracking functionality automatically

DATA COLLECTION: THE REWARD MAXIMIZATION PROBLEM

REWARD MAXIMIZATION MISSION



Node 4 "repelled" by Node 3

→ Search task performed

MISSION OBJECTIVE: *MAXIMIZE TOTAL REWARD COLLECTED BY VISITING TARGETS BEFORE THEIR "DEADLINES" EXPIRE*

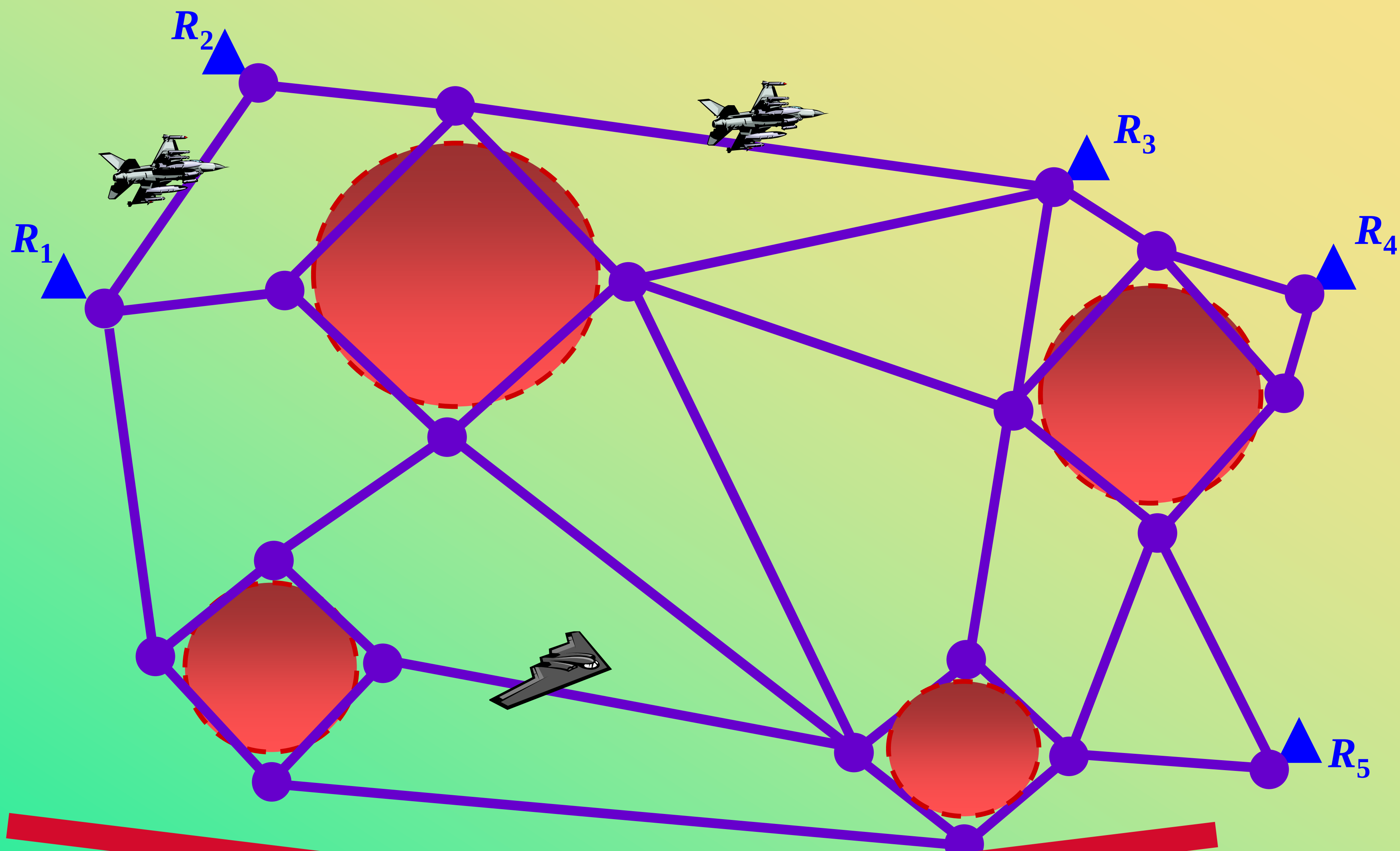
REWARD MAXIMIZATION MISSION

CONTINUED

This is like the notorious TRAVELING SALESMAN problem, except that…

- … there are multiple (cooperating) salesmen
- … there are deadlines + time-varying rewards
- … environment is stochastic
  (nodes may fail, threats damage nodes, etc.)

COMBINATORIAL + STOCHASTIC COMPLEXITY



~~1. Target Assignment → 2. Routing → 3. Path Control~~

THE BIGGER PICTURE: MANAGING UNCERTAINTY

UNCERTAINTY: CONTRAST TWO APPROACHES

ESTIMATE-AND-PLAN VS **HEDGE-AND-REACT**

ESTIMATE-AND-PLAN

- Decisions planned ahead
- Need accurate stochastic models
- Curse of dimensionality

- *Dynamic Programming (DP)*
- *Markov Decision Processes (MDP)*

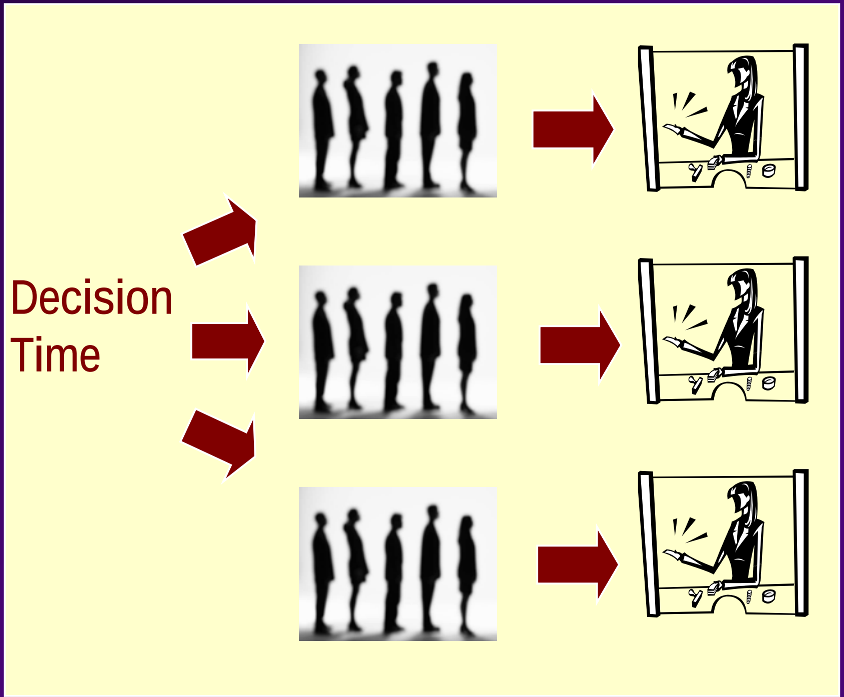
HEDGE-AND-REACT

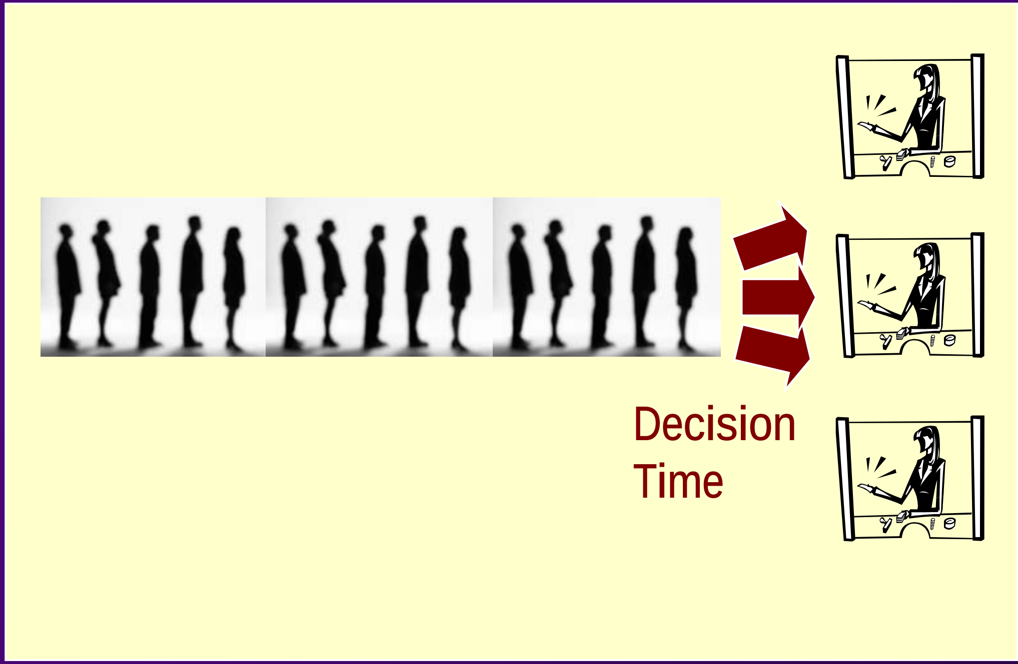
- Delay decisions until last possible instant
- No (detailed) stochastic model
- Simpler opt. problems

- *Receding Horizon Control (RHC)*
- *Model Predictive Control (MPC)*

UNCERTAINTY: CONTRAST TWO APPROACHES

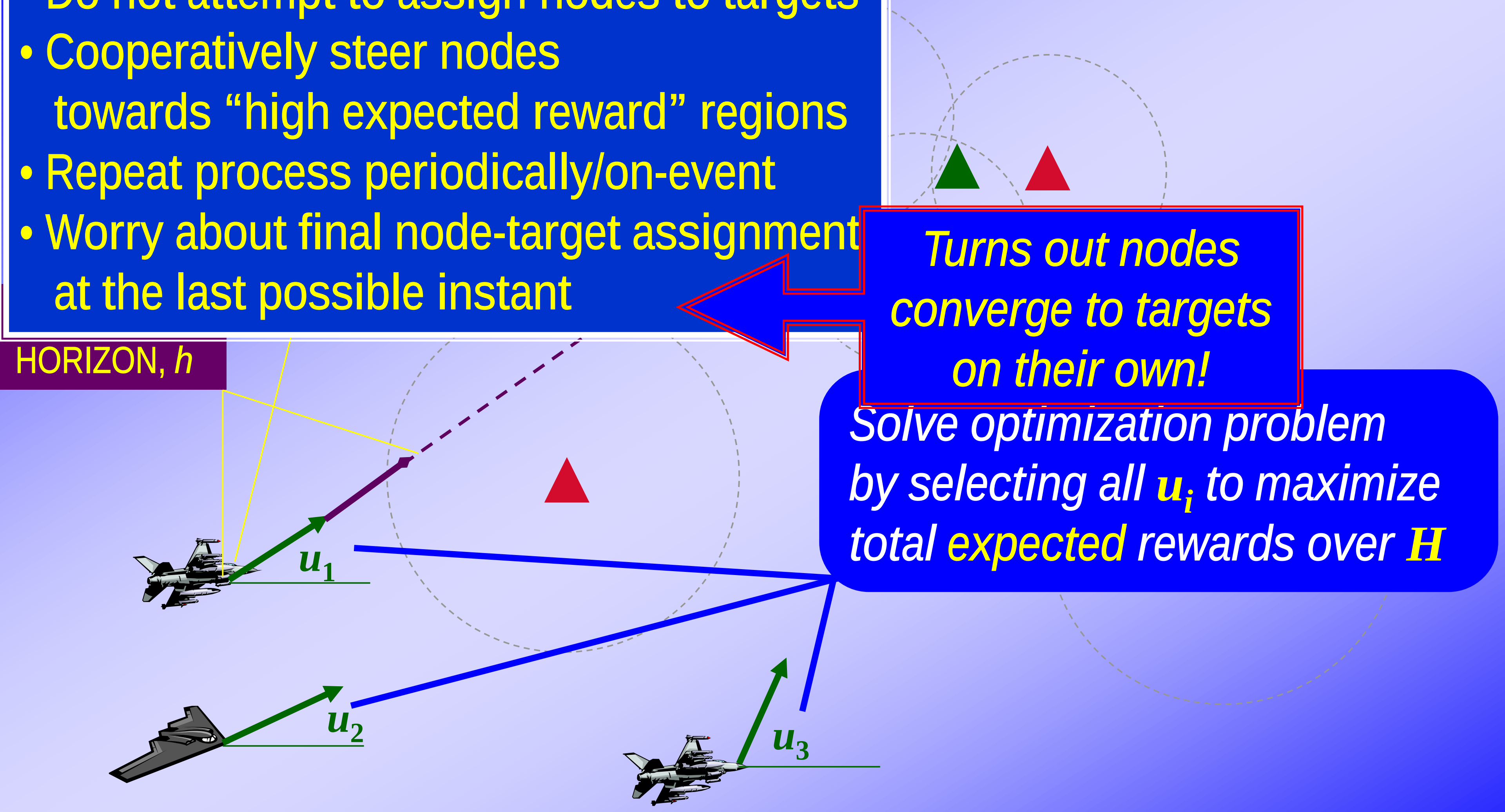
ESTIMATE-AND-PLAN VS HEDGE-AND-REACT





COOPERATIVE RECEDING HORIZON (CRH) CONTROL: *MAIN IDEA*

- Do not attempt to assign nodes to targets
- Cooperatively steer nodes towards "high expected reward" regions
- Repeat process periodically/on-event
- Worry about final node-target assignment at the last possible instant



TARGET ASSIGNMENT

MAIN IDEA IN CRH APPROACH:

Replace complex *Discrete Stochastic Optimization* problem by a sequence of simpler *Continuous Optimization* problems

But how do we guarantee that nodes ultimately head for the desired DISCRETE TARGET POINTS?

STABILITY ANALYSIS

- TARGETS: y_i
- NODES: x_j

DEFINITION: Node trajectory $\mathbf{x}(t) = \left[x_1(t), \ldots, x_M(t)\right]$ generated by a controller is *stationary*, if there exists some $t_V < \infty$, such that $\left\| x_j(t_v) - y_i \right\| \le s_i$ for some $i = 1, \ldots, N,\ j = 1, \ldots, M$.

Target Size

QUESTION:
Under what conditions is a CRH-generated trajectory stationary ?

MAIN STABILITY RESULT

Local minima of objective function $J(x)$: $x^l = (x_1^l, \ldots, x_M^l) \in \mathbf{R}^{2M}, \quad l = 1, \ldots, L$

Vector of node positions
at ***k***th iteration of CRH controller: $\mathbf{x}_k$

Theorem: Suppose $H_k = \min_{i,j} d_{ij}(t_k)$.
If, for all $l = 1,\ldots,L$, $x_j^l = y_i$ for some $i = 1,\ldots,N$, $j = 1,\ldots,M$,
then $J(\mathbf{x}_k) - J(\mathbf{x}_{k+1}) > b$ ($b > 0$ is a constant).

⇓

If all local minima coincide with targets,
the CRH-generated trajectory is stationary

MAIN STABILITY RESULT

QUESTION:

When do all local minima coincide with target points?

1 Node, *N* targets

If there exists a y_i s.t. $R_i - \left\| \sum_{j=1, j\neq i}^{N} R_j \frac{y_i - y_j}{\|y_i - y_j\|} \right\| > 0$

2 Nodes, 1 target

2 Nodes, 2 targets

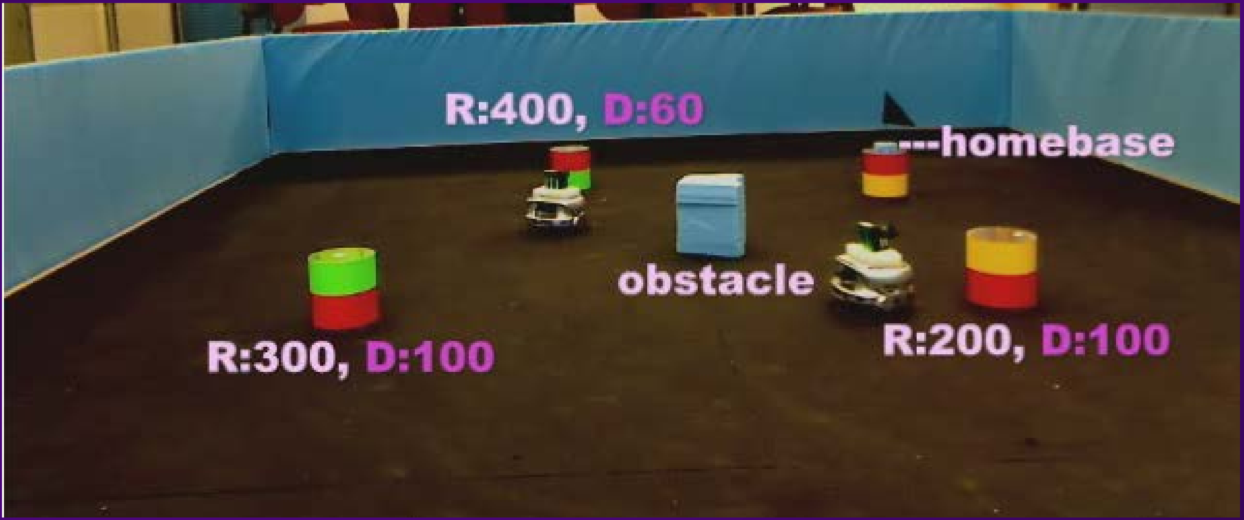
OTHER ISSUES

- Local optima in the CRH optimization problem
- Oscillatory node behavior (*instabilities*)
- Additional path constraints,
  e.g., rendez-vous at targets
- Does CRH control generate optimal assignments?

BOSTON UNIVERSITY TEST BEDS

RoboticUrban-like Environment (RULE)

CRH Test Bed with autonomous robots



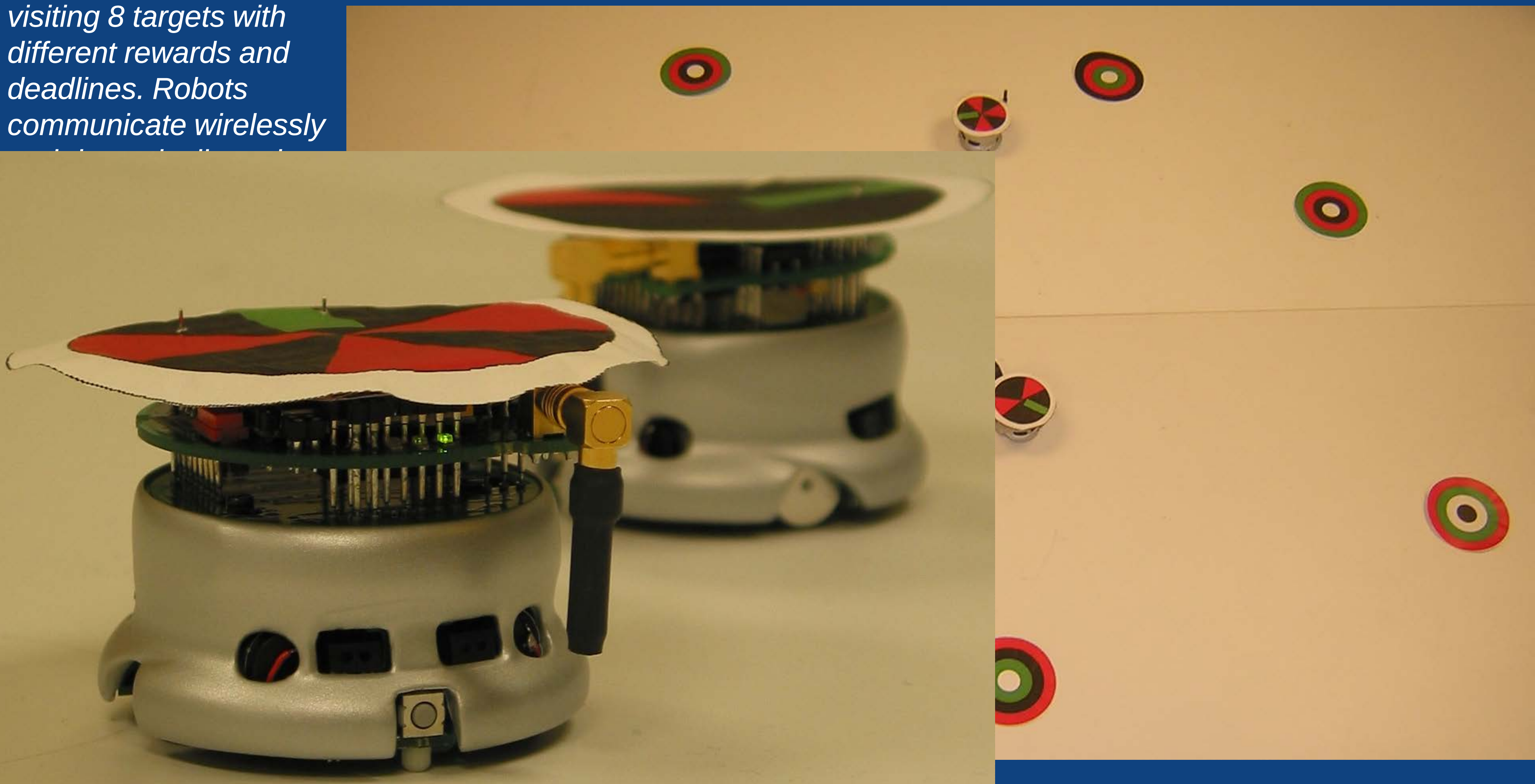
New autonomous robots

REWARD MAXIMIZATION DEMO

MOVIES OF SUCH PROBLEMS WITH SMALL ROBOTS:

http://codescolor.bu.edu/multimedia.html

3 Khepera robots executing mission: visiting 8 targets with different rewards and deadlines. Robots communicate wirelessly



BOSTON UNIVERSITY ROBOTIC URBAN TEST BED

SUMMARY, RESEARCH DIRECTIONS

- Small, cheap cooperating devices cannot handle complexity
  $\Rightarrow$ we need *DISTRIBUTED* control and optim. algorithms
- Cooperating agents operate autonomously (asynchronously)
  $\Rightarrow$ we need *ASYNCHRONOUS* (*EVENT-DRIVEN*) control/optimization schemes
- Too much communication kills node energy sources
  $\Rightarrow$ communicate ONLY when necessary
  $\Rightarrow$ we need *EVENT-DRIVEN* control/optimization schemes
- Networks grow large, sensing tasks grow large
  $\Rightarrow$ we need *SCALABLE* control and optim. algorithms

THRESHOLD PROCESS

$K_\delta > 0$

Update Direction, usually

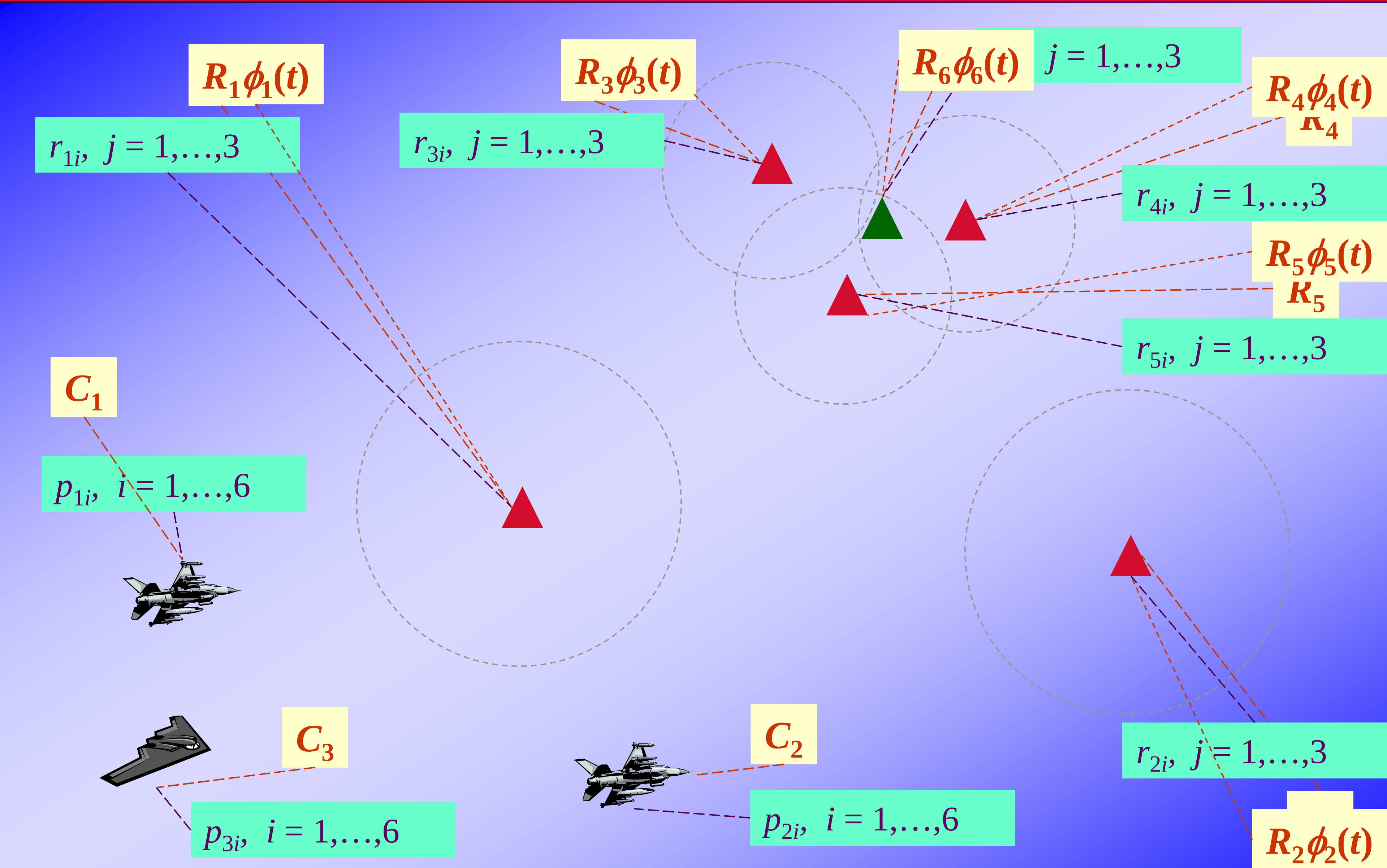
$d_i(\mathbf{s}^i(k)) = -\nabla_i H(\mathbf{s}^i(k))$

$$\delta_i(k) = \begin{cases} K_\delta \left\| d_i(\mathbf{s}^i(k) \right\| & \text{if } k \in C^i \\ \delta_i(k-1) & \text{otherwise} \end{cases}$$

$$\delta_i(0) = K_\delta \left\| d_i(\mathbf{s}^i(0) \right\|$$

Intuition:
near convergence
(small $d_i(\mathbf{s}^i(k))$),
better estimates are needed

COOPERATIVE *REWARD MAXIMIZATION* PROBLEM



SOLUTION APPROACHES

- Stochastic Dynamic Programming – *Wohletz et al, 2001*

  Extremely complex…

- Functional Decomposition:
  - Dynamic Resource Allocation – *Castanon and Wohletz, 2002*
  - Assignment Problems through Mixed Integer Linear Programming – *Bellingham et al, 2002*

    Combinatorially complex…
  - Path Planning – *Hu and Sastry, 2001, Lian and Murray 2002, Gazi and Passino, 2002, Bachmayer and Leonard, 2002*

CRH CONTROL PROBLEM FORMULATION

- Target positions ($i = 1,\ldots,N$): $y_i \in \mathrm{R}^2$

- Node dynamics ($j = 1,\ldots,M$):
  - State: $x_j(t) \in \mathrm{R}^2$ position of jth node at time t
  - Control: $u_j(t)$ Node heading at time t

$$\dot{x}_j(t) = V_j \begin{bmatrix} \cos u_j(t) \\ \sin u_j(t) \end{bmatrix}, \quad x_j(0) = x_j^0$$

- At *kth* iteration, time t_k ($k=1,2,\ldots$):
  - Planning Horizon: H_k
  - Node position at time t_k+H_k: $x_j(t_k + H_k) = x_j(t_k) + \dot{x}_j(t_k) H_k$

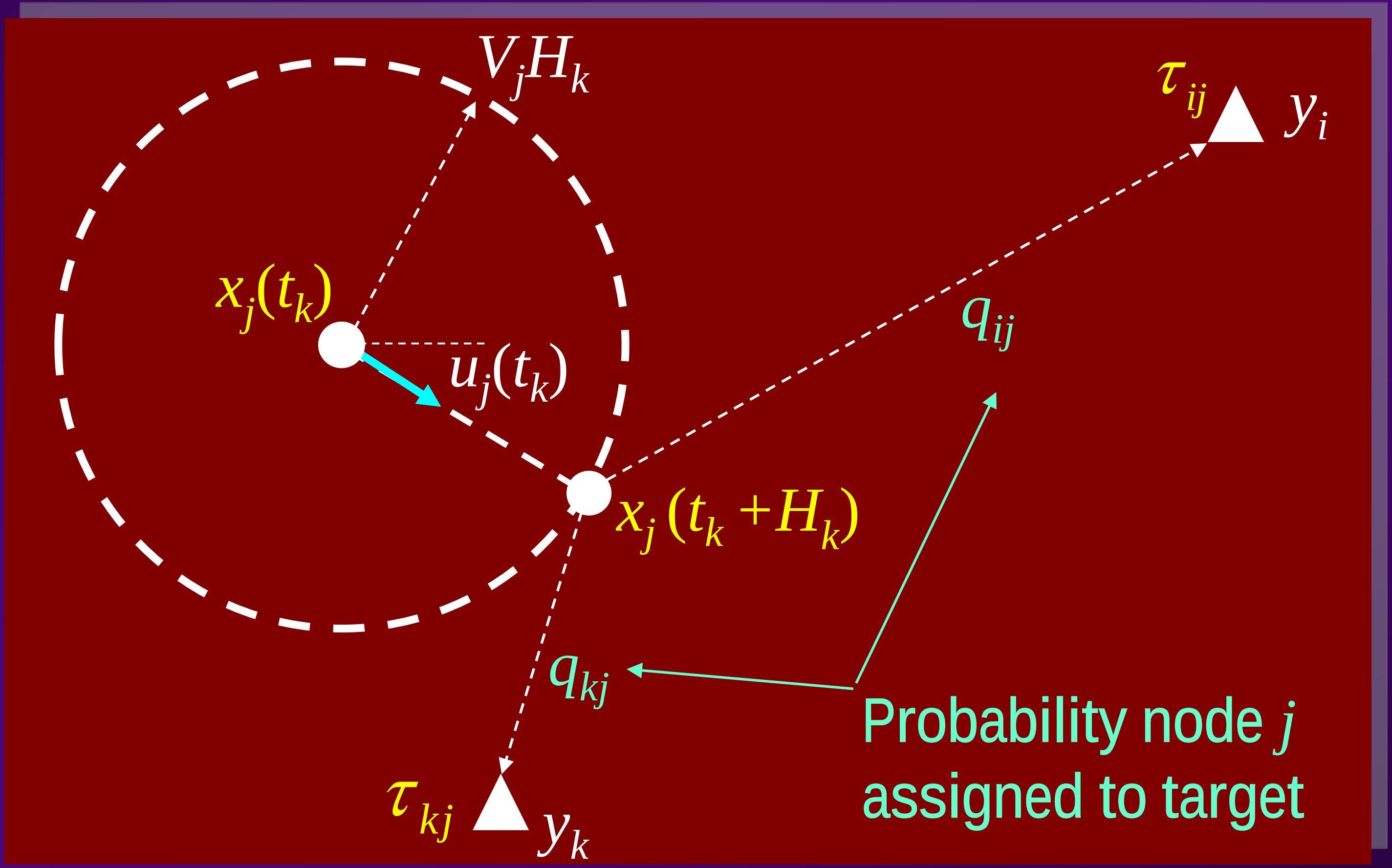
RH PROBLEM FORMULATION

CONTINUED

- At *kth* iteration (k=1,2,…):

Earliest time node j can reach target i under control $u_j(t_k)$:

$$\tau_{ij}(u_j(t_k),t_k) = (t_k + H_k) + \| x_j(t_k + H_k) - y_i \| / V_j$$



THE FUNCTION q_{ij} [TARGET ASSIGNMENT FUNCTION]

- Agent-to-target distance: $d_{ij} = \left\| x_j - y_i \right\|$
- Relative distance: $$\delta_{ij} = \frac{d_{ij}}{\sum_{m=1}^{M} d_{im}}$$

or: *b closest agents to j only*

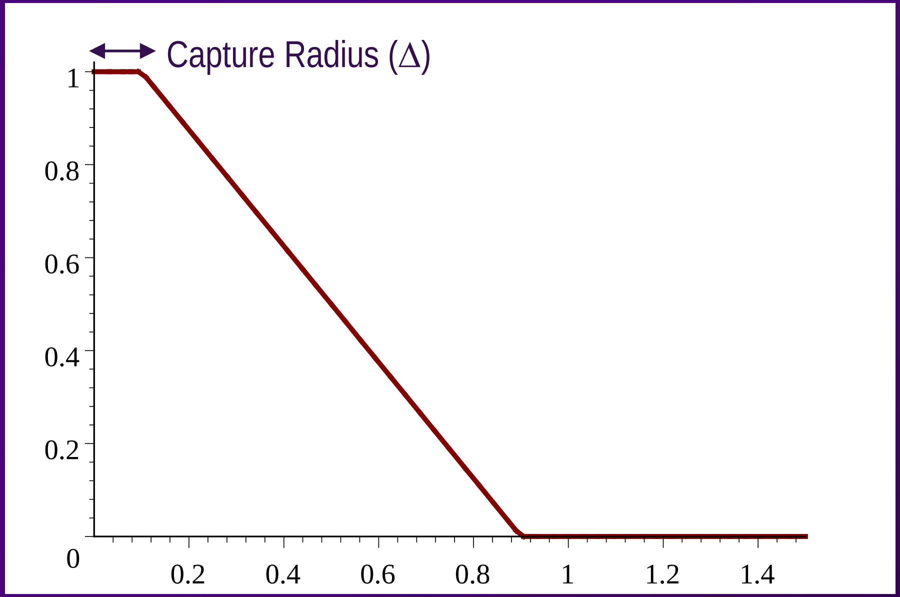
- Target assignment function $q_{ij}(\delta_{ij})$:

Monotonically non-increasing and s.t.

$$q_{ij}(0) = 1, \quad q_{ij}(1) = 0$$

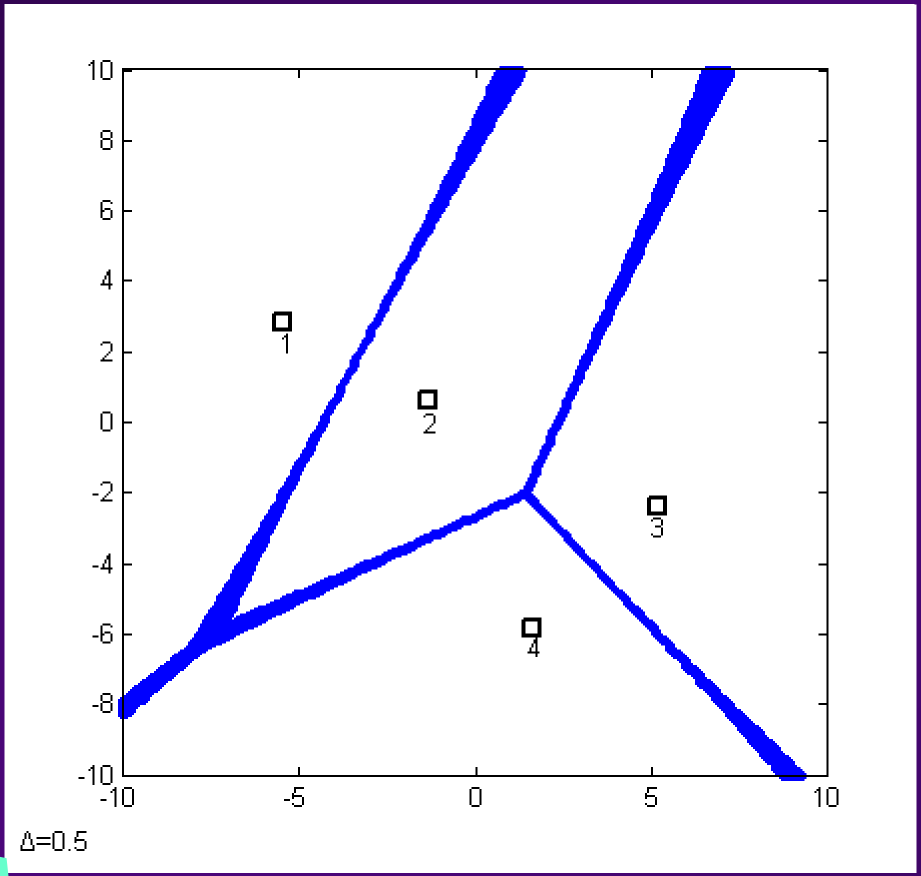
- A example of q_{ij} function $(M=2)$:

$$q_{ij}(\delta_{ij}) = \begin{cases} 1 & \text{if } \delta_{ij} \le \Delta \\ \dfrac{1}{1-2\Delta}\left[(1-\Delta)-\delta_{ij}\right] & \text{if } \Delta < \delta_{ij} \le 1-\Delta \\ 0 & \text{otherwise} \end{cases}$$



THE FUNCTION q_{ij}

CONTINUED



Voronoi partition

Partition of a plane with into n convex polygons such that each polygon contains exactly one point and every point in a given polygon is closer to its central point than to any other.

What happens as parameter Δ increases ?

CRH PROBLEM FORMULATION

CONTINUED

- Objective at kth iteration:

Maximize *EXPECTED REWARD* over horizon H_k

Target i value attainable by node j
[depends on $u_j(t)$]

$$\max_{\mathbf{u}} \sum_{i=1}^{M}\sum_{j=1}^{N} R_i\phi_i(\tau_{ij})\, p_{ij}(\tau_{ij})\, q_{ij}(t_k+H_k)$$

Control node headings

Prob. node j collects target i reward [depends on $u_j(t)$]

Earliest time when node j can collect reward from target i
[depends on $u_j(t)$]

Prob. node j assigned to target i
[depends on $u_j(t)$]

PLANNING AND ACTION HORIZONS

PLANNING Horizon ***H(t)***:

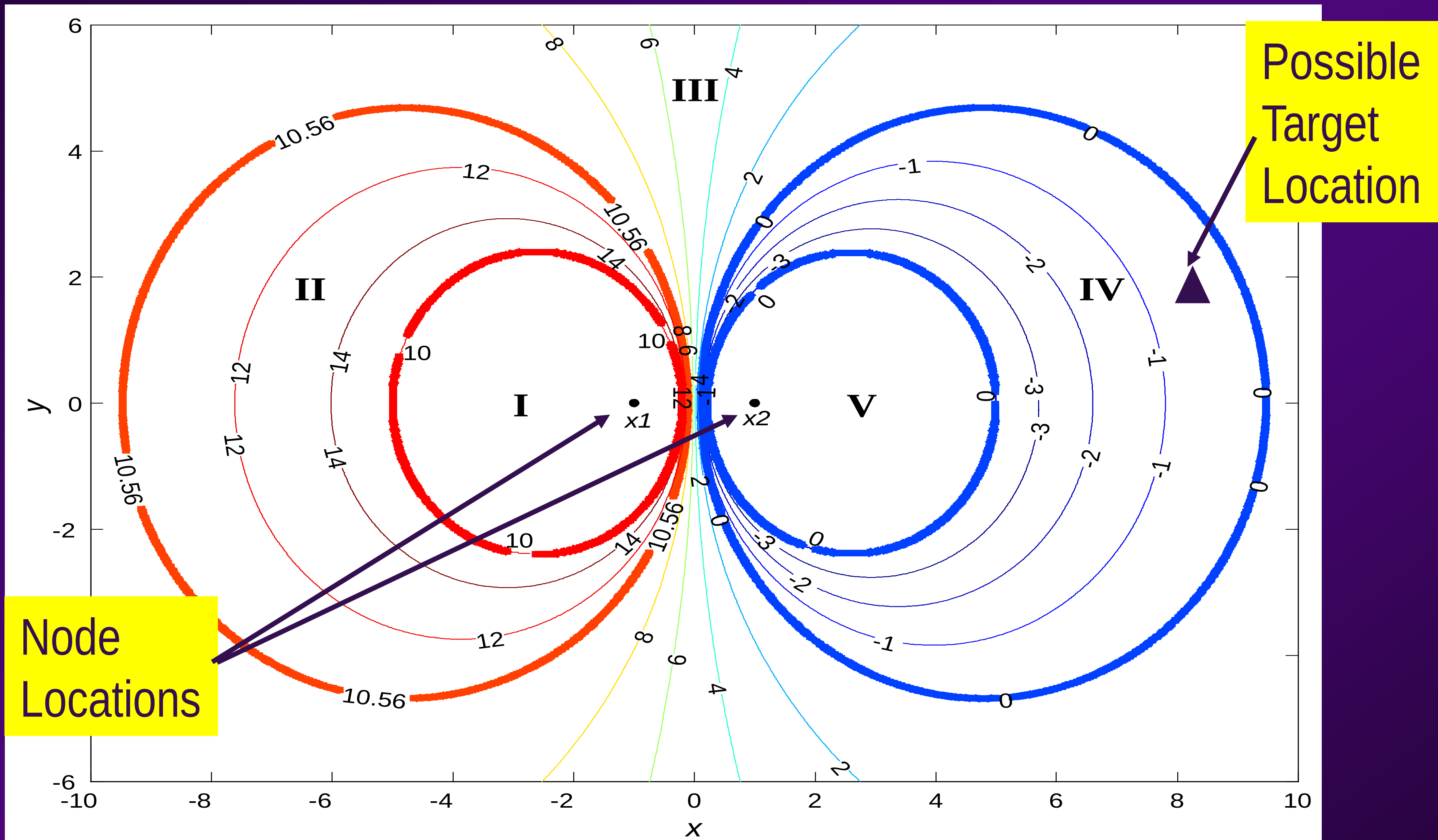
$$H(t) = d_{\min}(t) \equiv \min_{i,j} d_{ij}(t)$$

ACTION Horizon ***h(t)***:

$$h(t) = \alpha_H + \beta_H H(t), \quad \alpha_H \geq 0, \quad 0 \leq \beta_H \leq 1$$

OR: Whenever next EVENT occurs

2-NODE CASE – DYNAMIC PARTITIONING



II: Only node 1 goes to target

III: Both nodes go to target

IV: Only node 2 goes to target (1 is repelled !)