Surveillance strategies for autonomous mobile robots

Nicola Basilico
Department of Computer Science
University of Milan

Intelligence, surveillance, and reconnaissance (ISR) with autonomous UAVs

- ISR defines a class of real world applications where robots can be effectively employed
- UAVs are an emerging technology in this field:
  - Large number of real world deployments
  - High level autonomy is an open challenge
- Our reference application: cooperative surveillance by a team autonomous UAVs
- Key feature: robots have faulty sensors to do detect attacks

Background and motivations

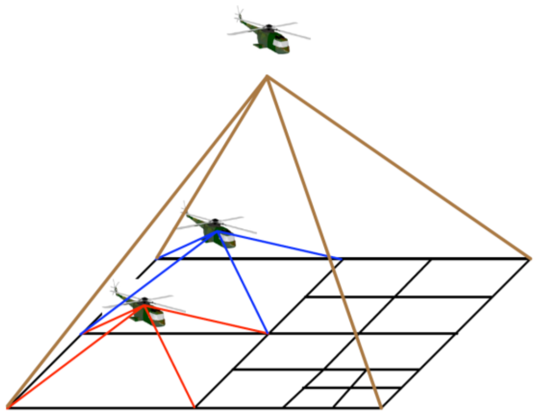
- One central problem in autonomous surveillance is the definition of a **surveillance strategy**: how to schedule environmental inspections in space and time
- It's an online problem: where to inspect next in general depends on what has been seen so far
- It has theoretical roots in search theory [Koopman, 1956], [Stone, 2007]
- It needs for practical and scalable methods that can cope with features of real world robotic settings

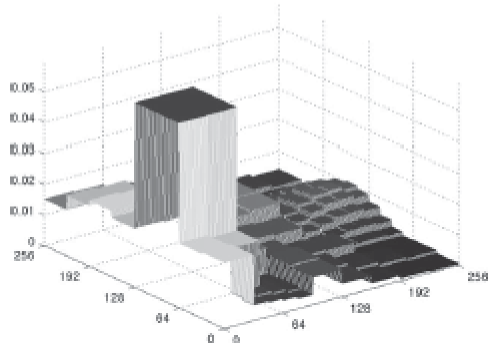
Decision theory

Game theory

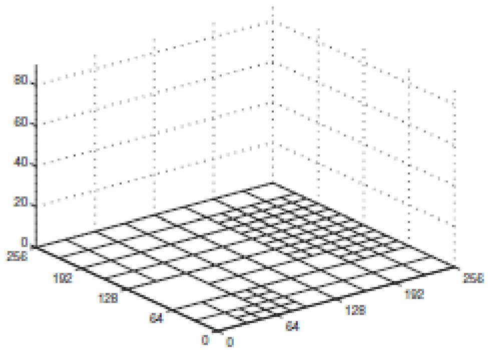
Background and motivations

- Variable resolution environment representations [Carpin et al 2011, 2012]:
  - Robots use a multi-scale, dynamically maintained, quadtree representation of the environment
  - More efficiency with small losses of performance





(b) PQ prior with 150 nodes



(c) Quadtree with 150 nodes

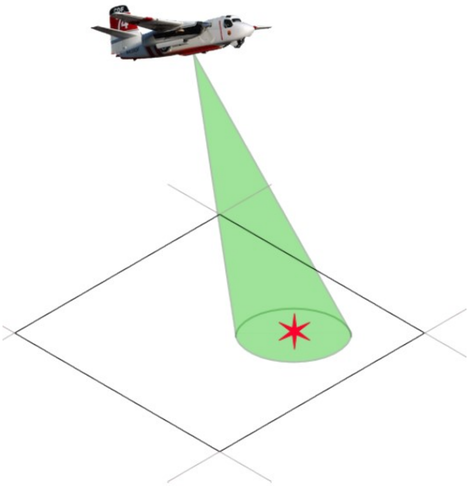
Background and motivations

- Two-level robot coordination [Basilico et al., 2014]:
- UAVs are assigned roles:
  - Sentinels undertake broad-area surveillance
  - Searchers perform triggered local inspections

Background and motivations

- Two-level robot coordination [Basilico et al., 2014]:

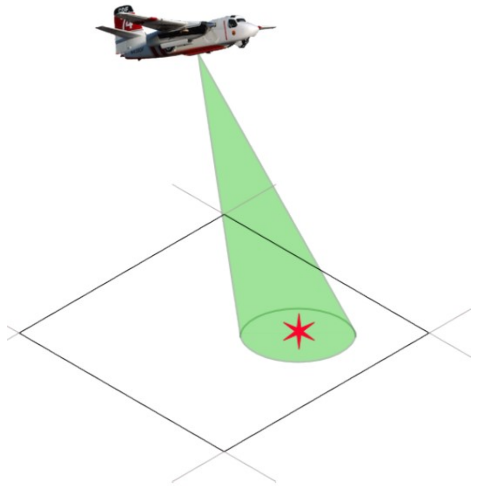
- UAVs are assigned roles:
  - Sentinels undertake broad-area surveillance
  - Searchers perform triggered local inspections



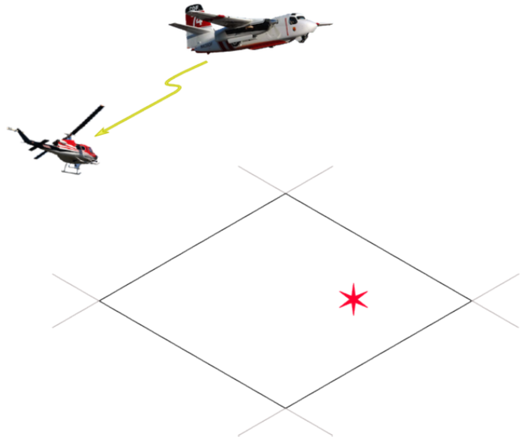
Broad area surveillance:
detection

Background and motivations

- Two-level robot coordination [Basilico et al., 2014]:
- UAVs are assigned roles:
  - Sentinels undertake broad-area surveillance
  - Searchers perform triggered local inspections



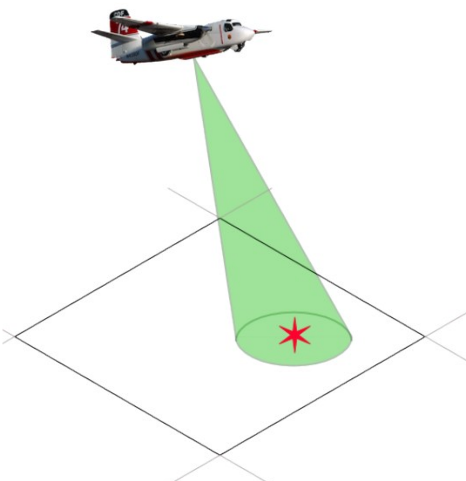
Broad area surveillance: detection



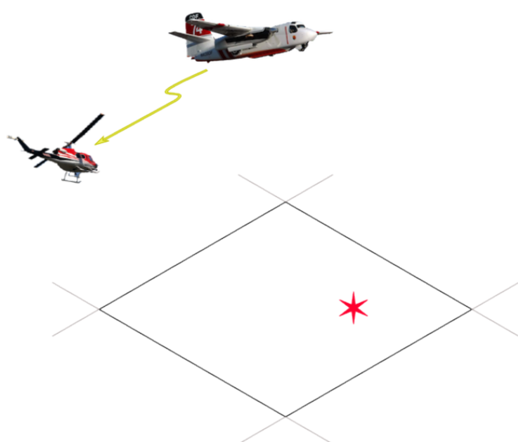
Broad area surveillance: dispatch

Background and motivations

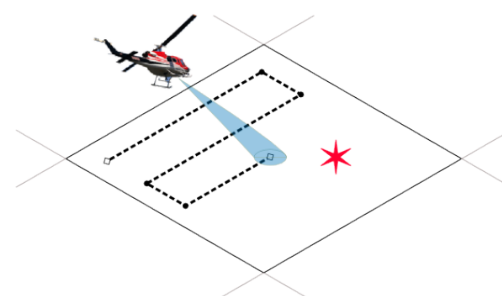
- Two-level robot coordination [Basilico et al., 2014]:
- UAVs are assigned roles:
  - Sentinels undertake broad-area surveillance
  - Searchers perform triggered local inspections



Broad area surveillance: detection

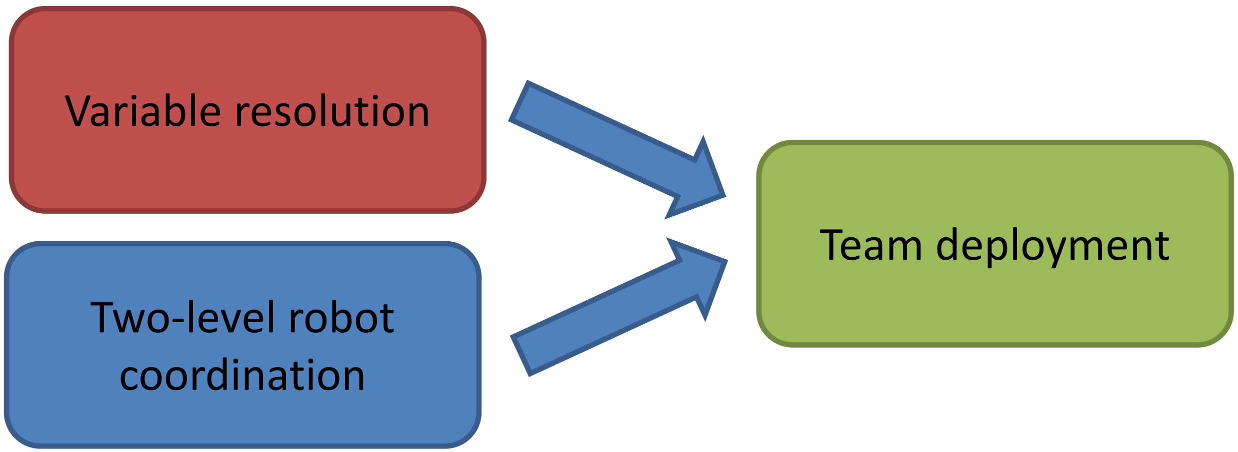


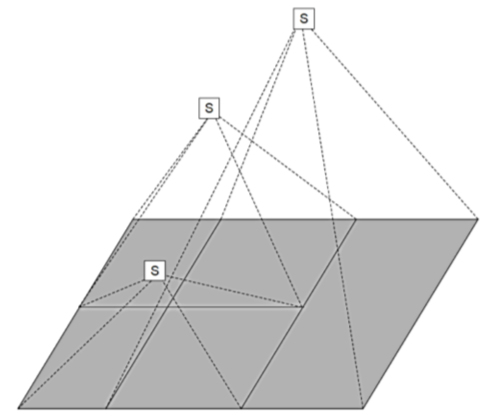
Broad area surveillance: dispatch



Local inspection

Background and motivations





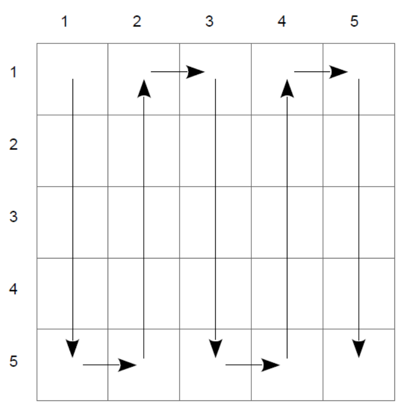
- Assign each sentinel a sub-area of the environment with the objective of:
  - Better distribute the common effort over the whole environment
  - Exploit synergies given by overlaps in the most sensitive regions
- Problem 1: given a candidate deployment, how to measure its goodness?
- Problem 2: how to efficiently search for an optimal deployment?

Evaluating deployments

- Environment discretized in cells, each cell c has a loss l(c) (the higher the loss the higher the damage per time unit of having an attack there)

- Attacks follow a probabilistic behavior
  - Spatial: proportional to the loss
  - Temporal: exponential arrival times

- Once dispatched, a searcher will perform a “lawn mower” pattern over its assigned sub-area



If sentinel is placed at position s, the expected total loss from a cell c is

$$v_s(c) = \mathbb{E}\left[l(c) \int_0^T a(c,t)dt \right]$$

Mission time

Loss of cell c

Attack function: equal to 1 if cell c is attacked at time t (unknown!)

Evaluating deployments

- We can provide an upper bound to the expected loss by combining the following expected quantities:
  - Time between attack arrival and scan in that cell
  - Num. scans to be performed before dispatching a searcher
  - Num. of searchers dispatches to be performed before the successful one
  - Time to detect the attack by the successful searcher

$$v_s(c) \leq W_c\Big(\mathbb{E}[\zeta] + \mathbb{E}[t_{tr}^1] + (\mathbb{E}[n_s] - 1)\mathbb{E}[t_{tr}] + \mathbb{E}[s]\Big)$$

- Each cell c can be covered by multiple sentinels: we take the tightest UB (min)

- The whole performance in the environment P is given by the largest UB (max)

- We aim at minimizing P

Searching for optimal deployments

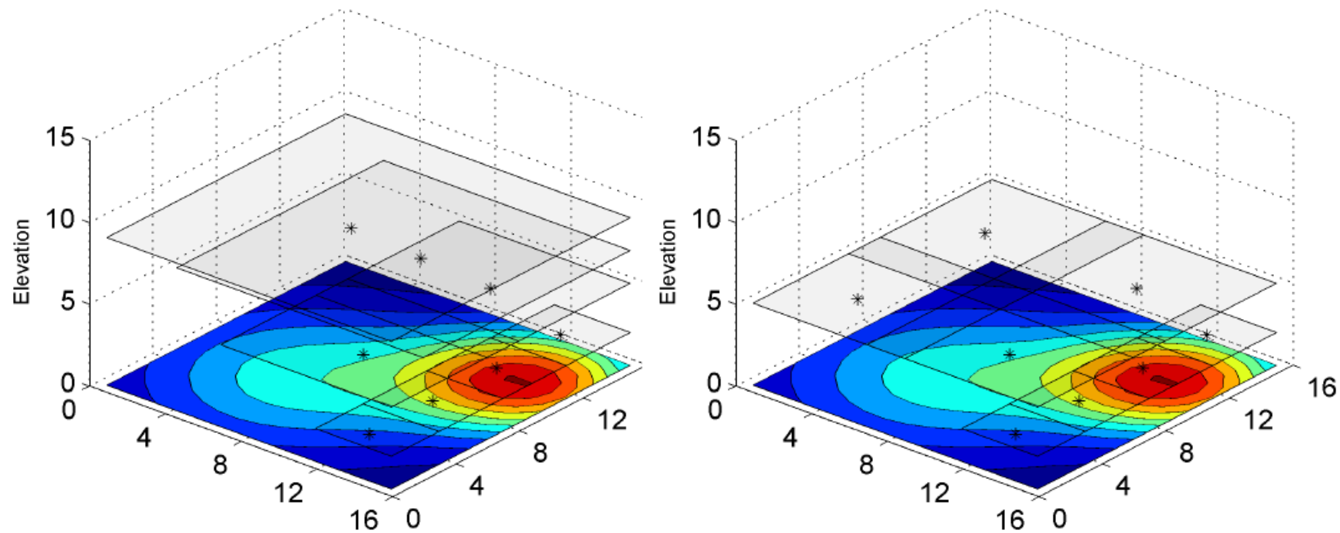
- We devised an iterative algorithm searching in the possible space of feasible deployments for M sentinels

- Tradeoff between exhaustive search ($\rho = 1$) and greedy construction ($\rho = M$)

- Contrarily to what intuition would suggest submodularity cannot be exploited to provide quality guarantees to the greedy method

Results

Example with 8 sentinels (red regions have higher losses)

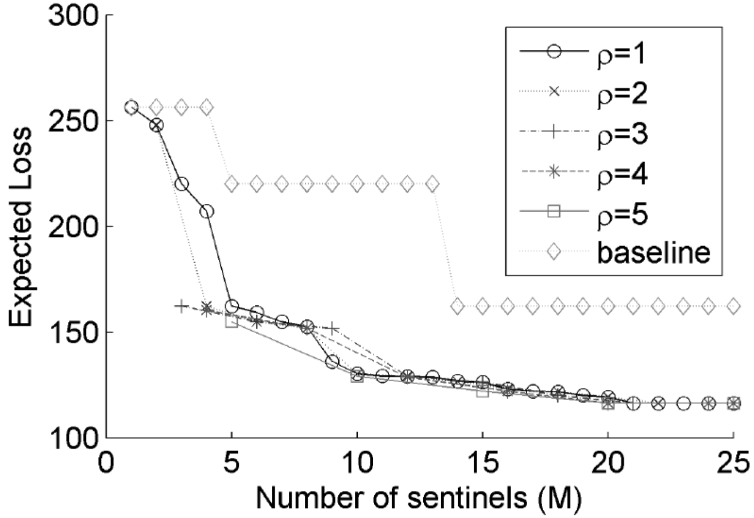


(a) Greedy deployment

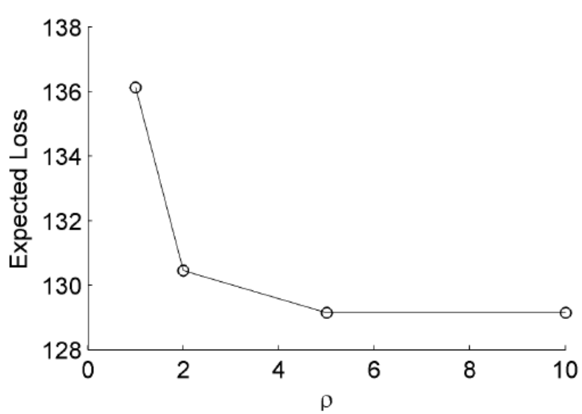
(b) Optimal deployment

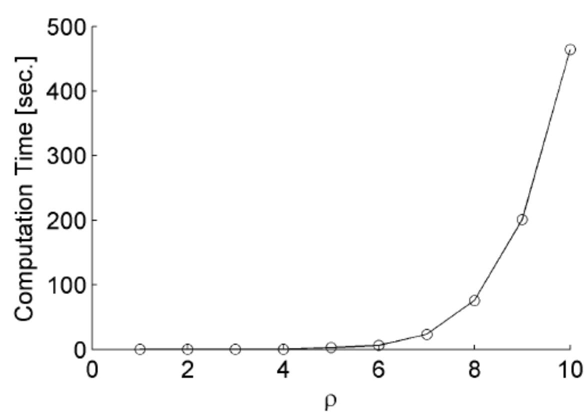
The optimal deployment better exploits overlap of different sentinels, also sentinels occupy lower positions (less energy required)

Results



Loss reduction as the number of sentinels increases: the choice of deployment can be critical





Quality vs computation time tradeoff of our method

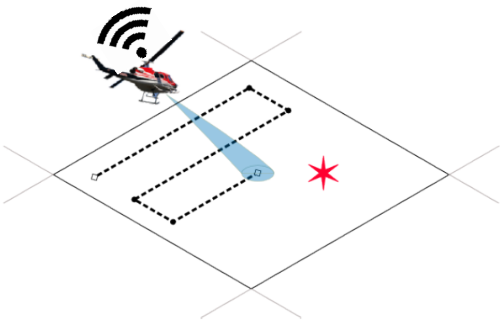
Remarks

- Team deployment is a critical issues besides the definition of good surveillance strategy

- Seeking effective and informed effort distribution can provide advantages during the on-line execution of surveillance missions

Surveillance in environments with restricted communication

- Searchers must promptly report what they find



- In communication-restricted environments some works (e.g.,[Mosteo et al., 2009, DARS]) try to maintain a multihop network

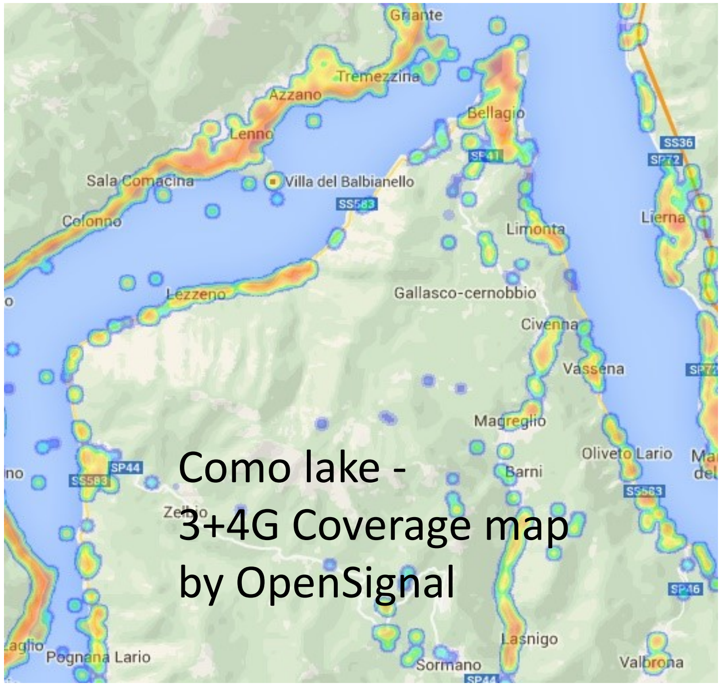
Surveillance in environments with restricted communication

Idea: exploit an existing communication infrastructure ([Tortonesi et al., 2012, IEEE Commun Mag] , [Ochoa and Santos, 2015, Inform Fusion]) providing partial coverage to the environment to make reports to a Mission Control Center (MCC)

Given some locations to monitor, two objectives:

1) Maximize the frequency of visit to the locations

2) Receive periodic and fresh reports at the MCC



Como lake -
3+4G Coverage map
by OpenSignal

Situation Aware Patrolling (SAP)

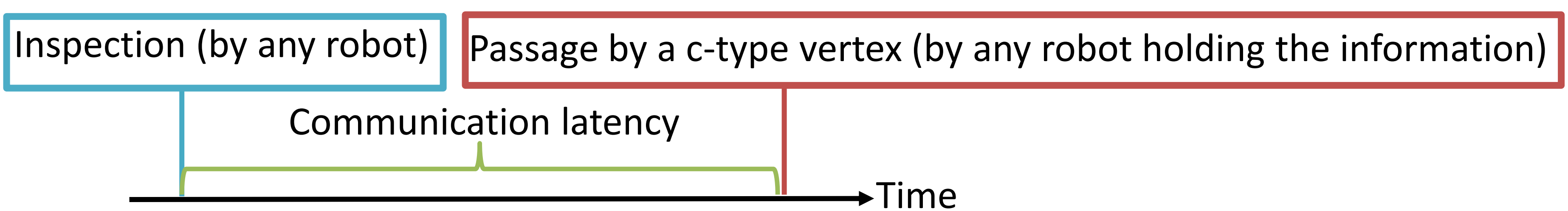
- Given a discretized environment G=(V,E) (weighted, metric)
  - V_m: locations to be monitored
  - V_c : locations with available communication

and:

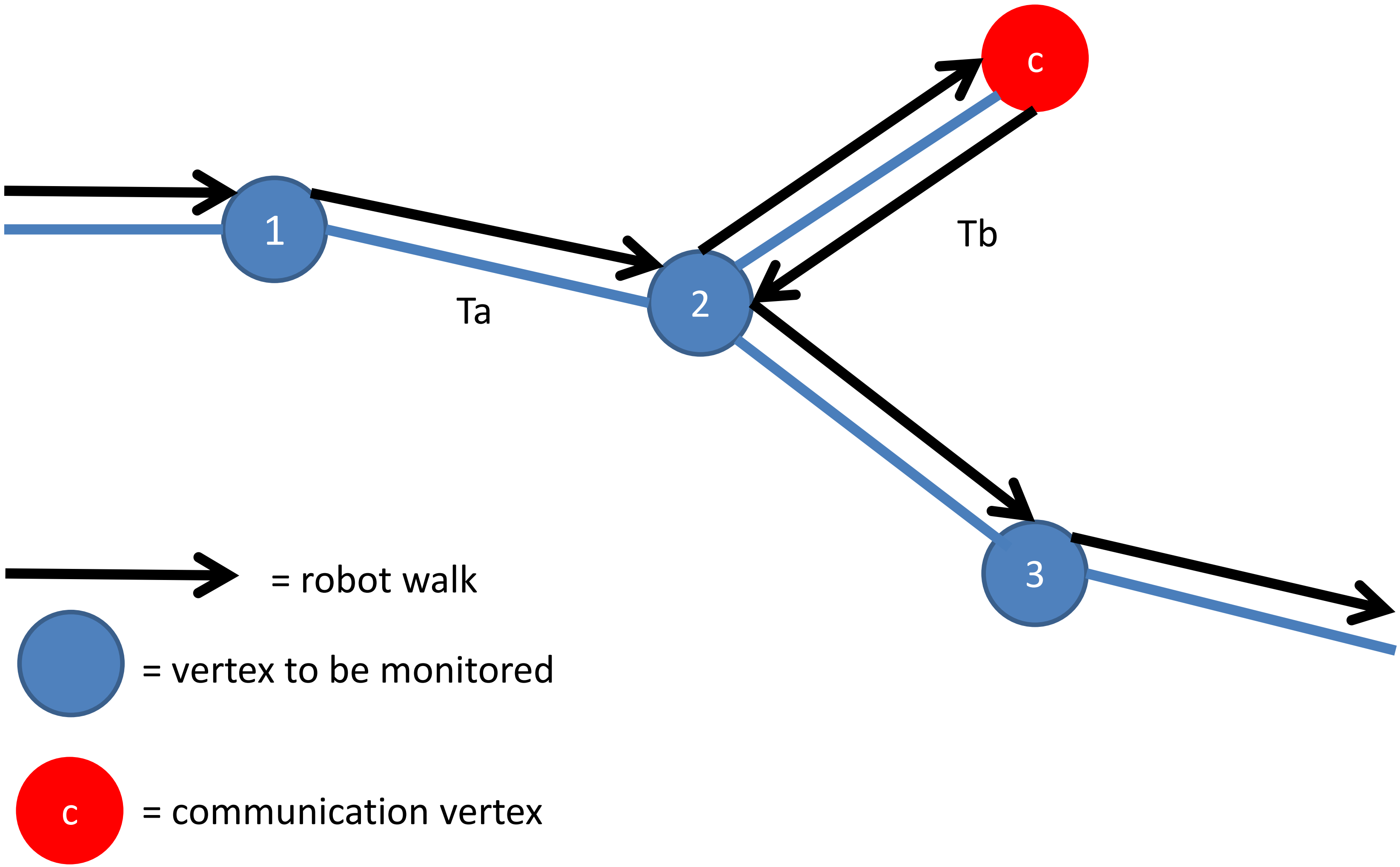
  - K UAVs
  - a time budget T
  - a starting depot d $\in V_c$

NP-hard and inapproximable

- Determine K cyclic walks of length $\leq$T on G minimizing the **average communication latency** of *m-type* vertices.



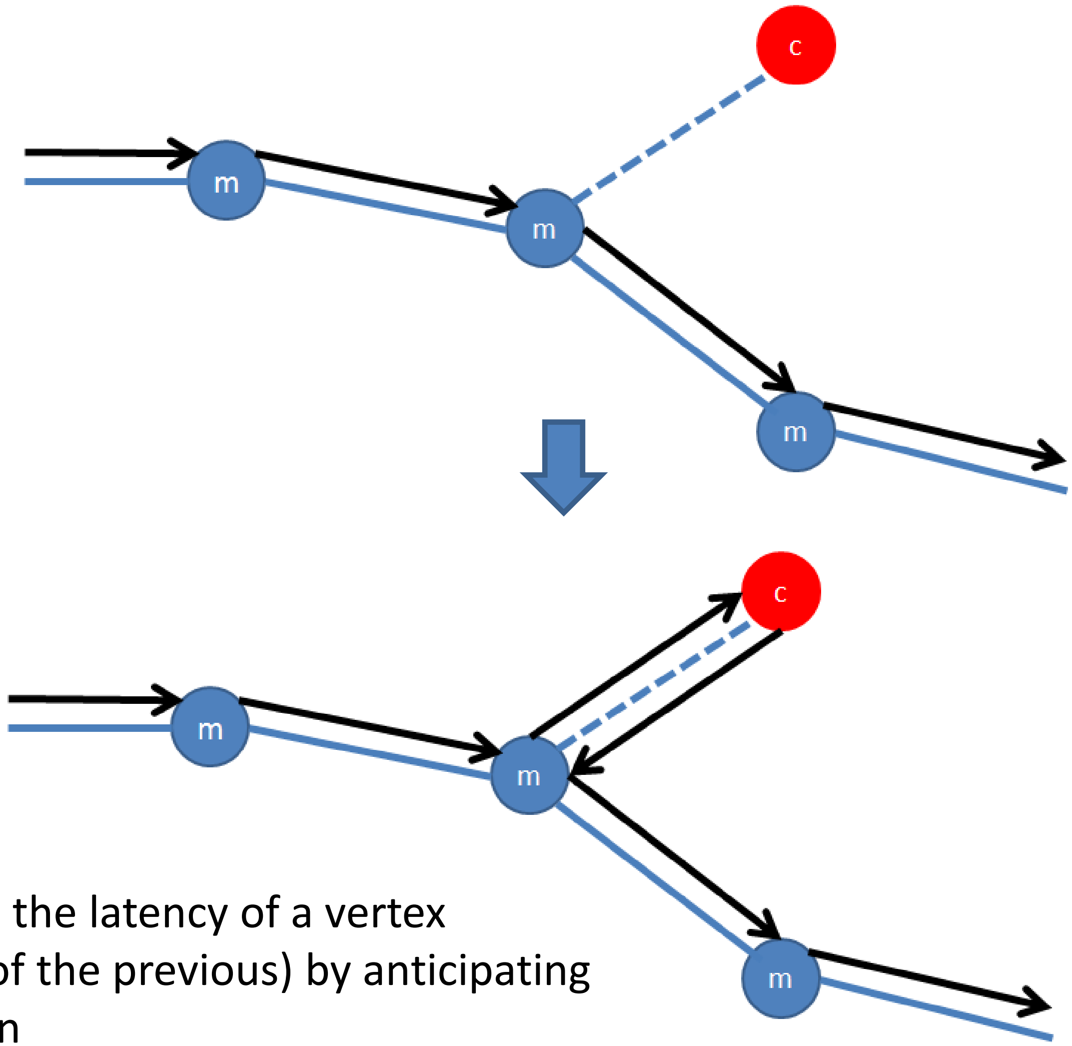
Simple example



Resolution methods

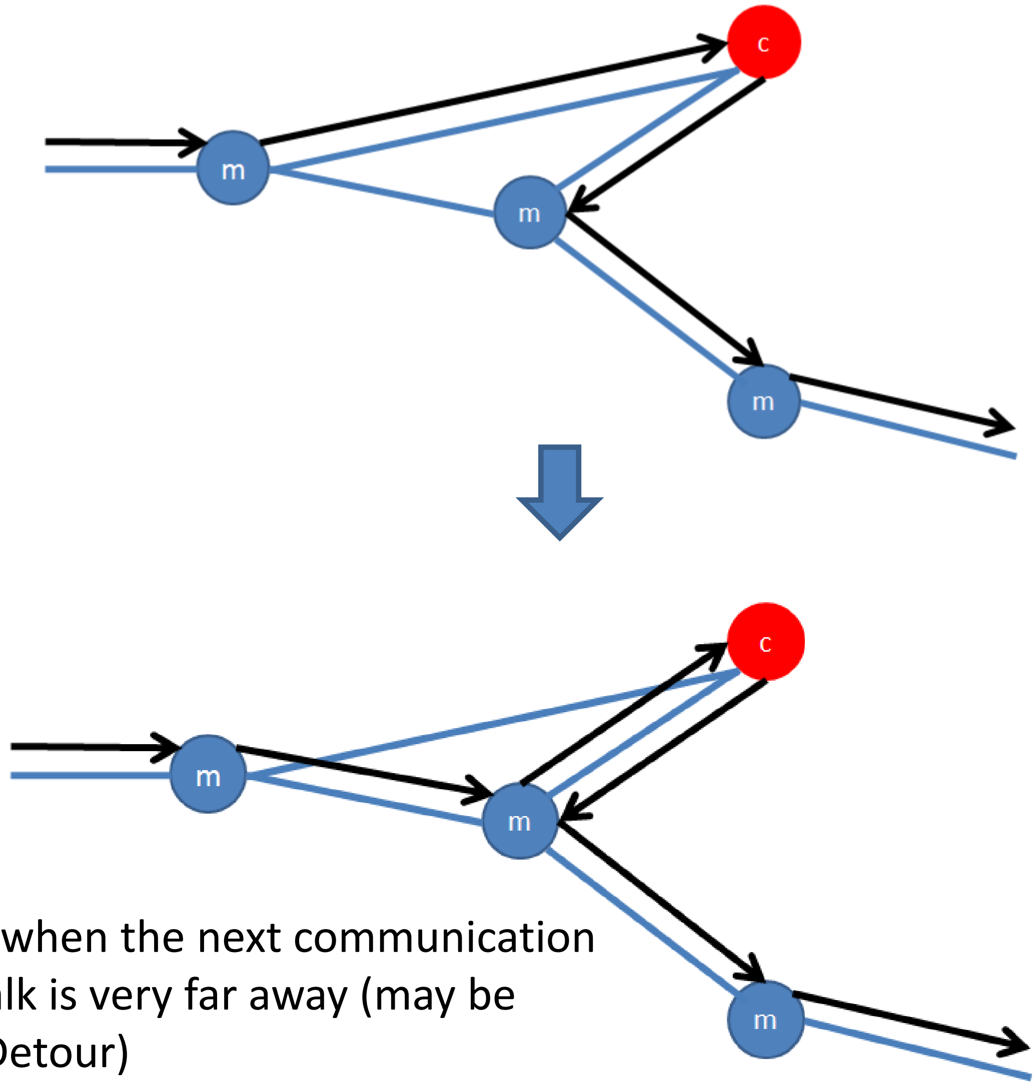
- Heuristic algorithm:
  - Start from a feasible solution of K walks (e.g., use Frederickson’s heuristic for the k-TSP) consuming as few budget as possible
  - Iteratively apply local modifications until some budget is available

Modifications - Detour



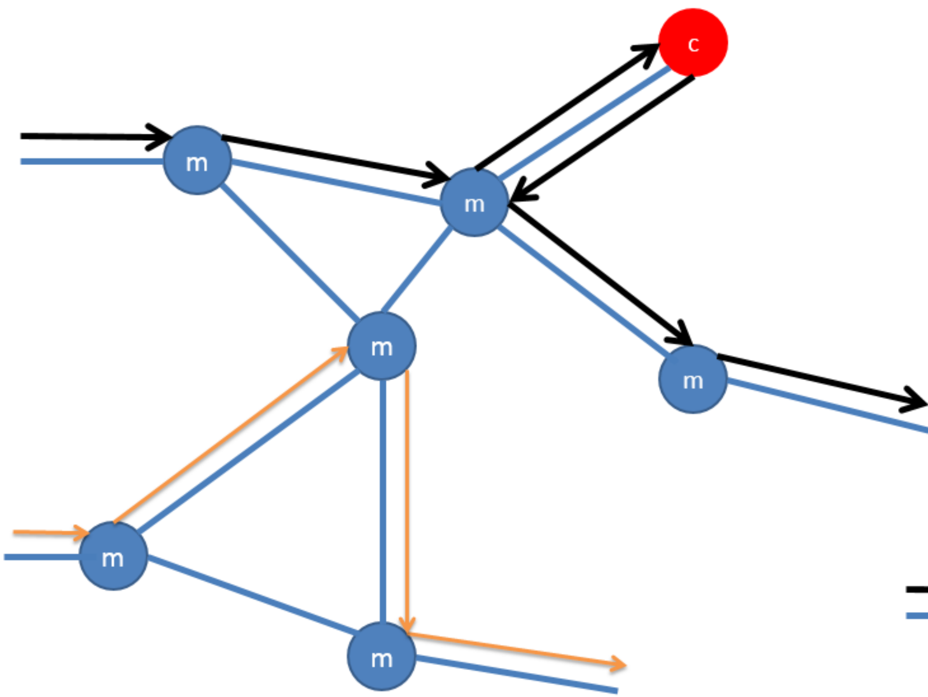
Idea: decrease the latency of a vertex
(and possibly of the previous) by anticipating
communication

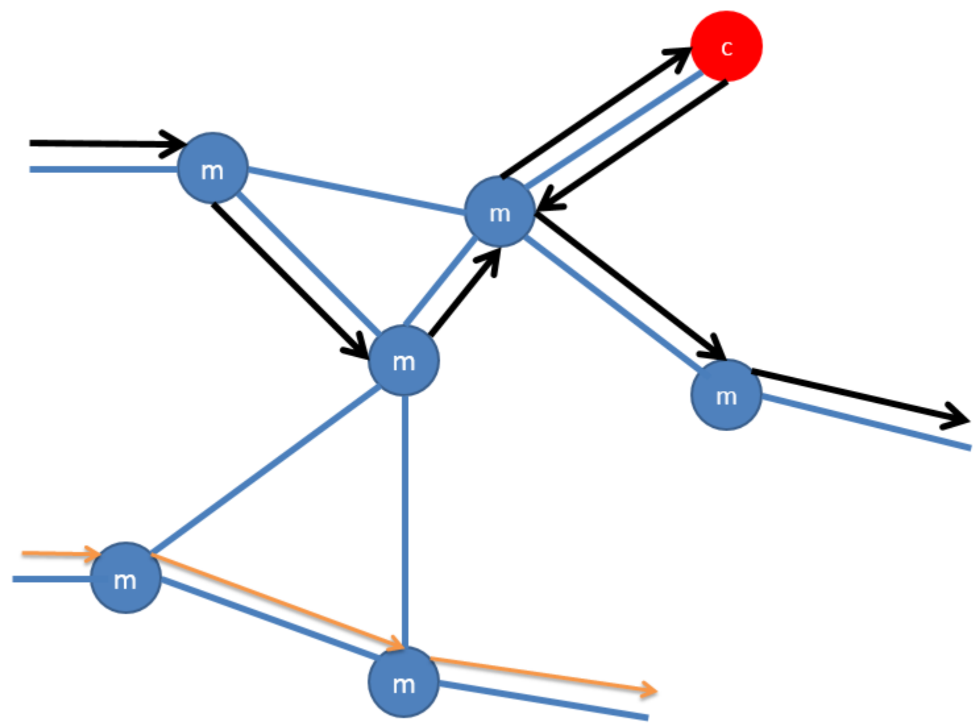
Modifications - Shift



Idea: effective when the next communication node in the walk is very far away (may be better than a Detour)

Modifications - Overlap





Idea: useful when robots start to run out of time budget

Heuristic algorithm

- Start from a feasible solution
- Repeat:
  - Construct all the possible modifications
  - Take the modification with the highest (improvement in latency)/(cost) ratio
  - Clean the solution (“Shortcut” modification removes useless portions of walks to regain some budget)
- Until some budget is available

Some results

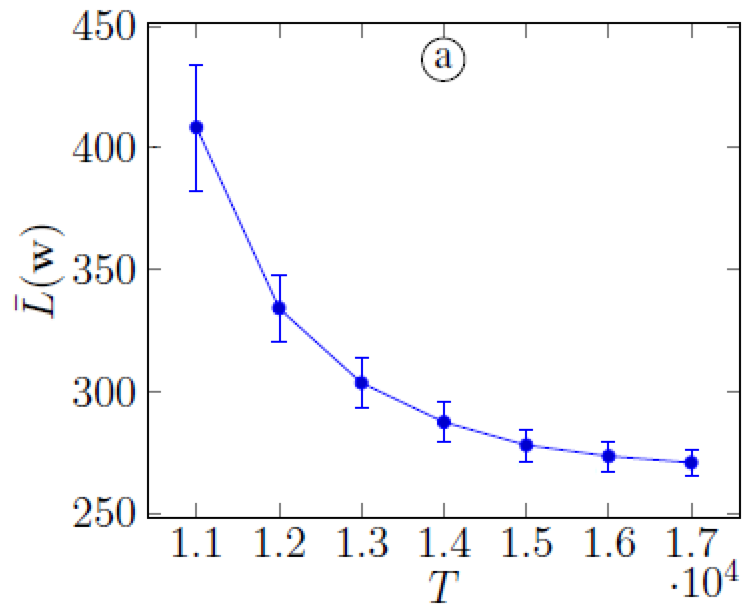
- Effect of local modifications in percentage (100 random graphs of 100 vertices on a 1000x1000 area, 4 robots)

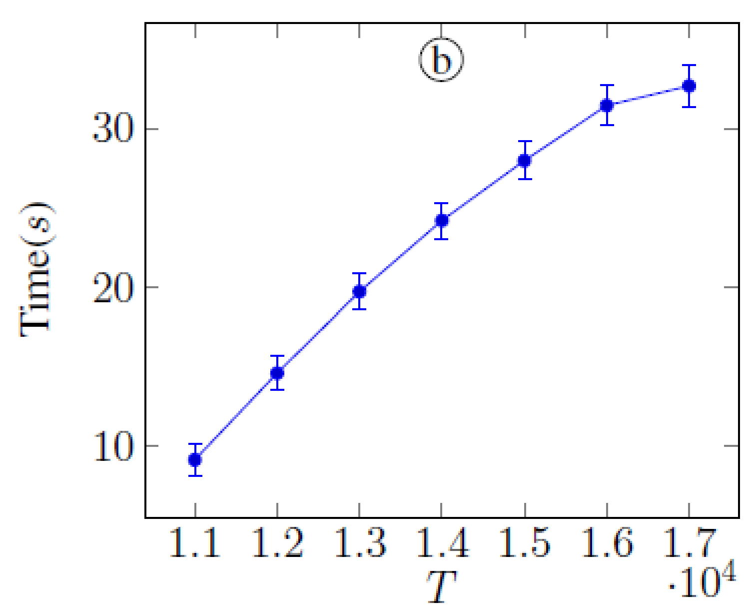
ε = graph edge density

ϵ	Detour	Shift	Overlap	Shortcut	Overall	
0.1	2.9 ±2.9	1.6 ±1.6	1.7 ±1.6	1.8 ±1.7	62 ±7	T = 13000
0.3	2.5 ±2.8	1.7 ±1.3	1.7 ±1.6	1.8 ±2.0	68 ±5	T = 8000
0.5	2.4 ±2.7	1.6 ±1.3	1.6 ±1.6	1.8 ±2.0	70 ±5	T = 7000
0.7	2.4 ±2.8	1.5 ±1.2	1.5 ±1.6	1.9 ±1.8	71 ±5	T = 5000

Some results

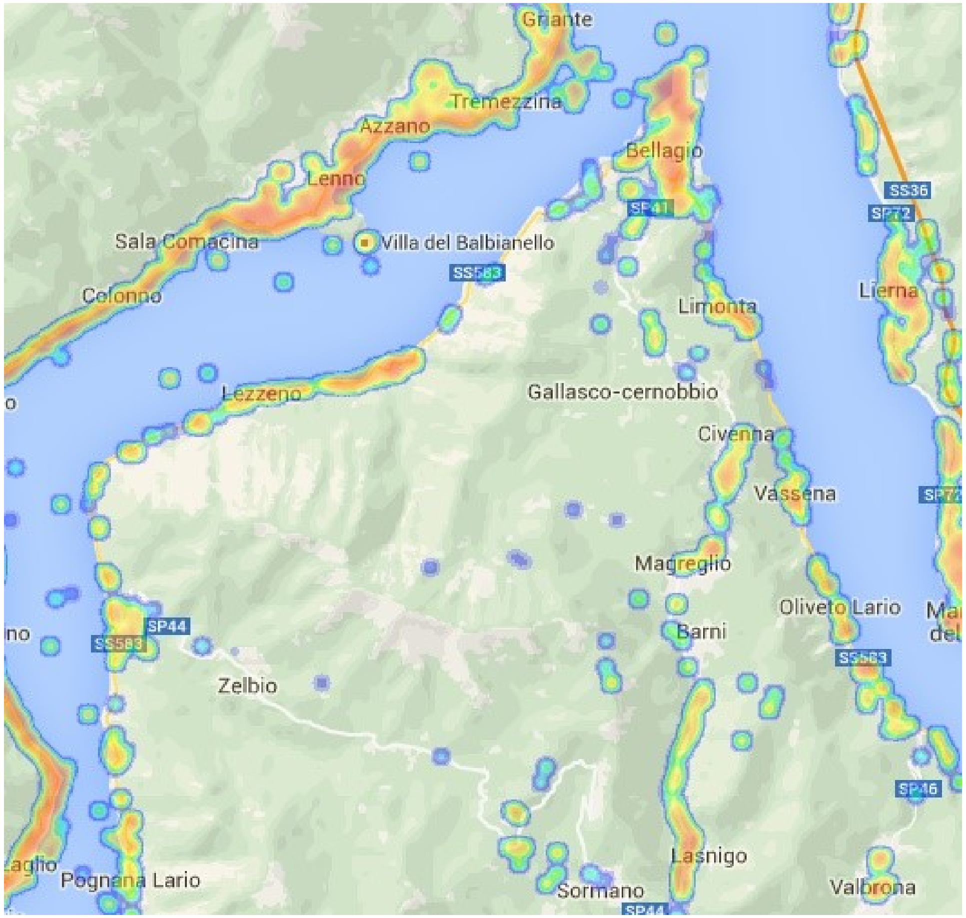
- Increasing the time budget ($\varepsilon = 0.1$, 4 robots)





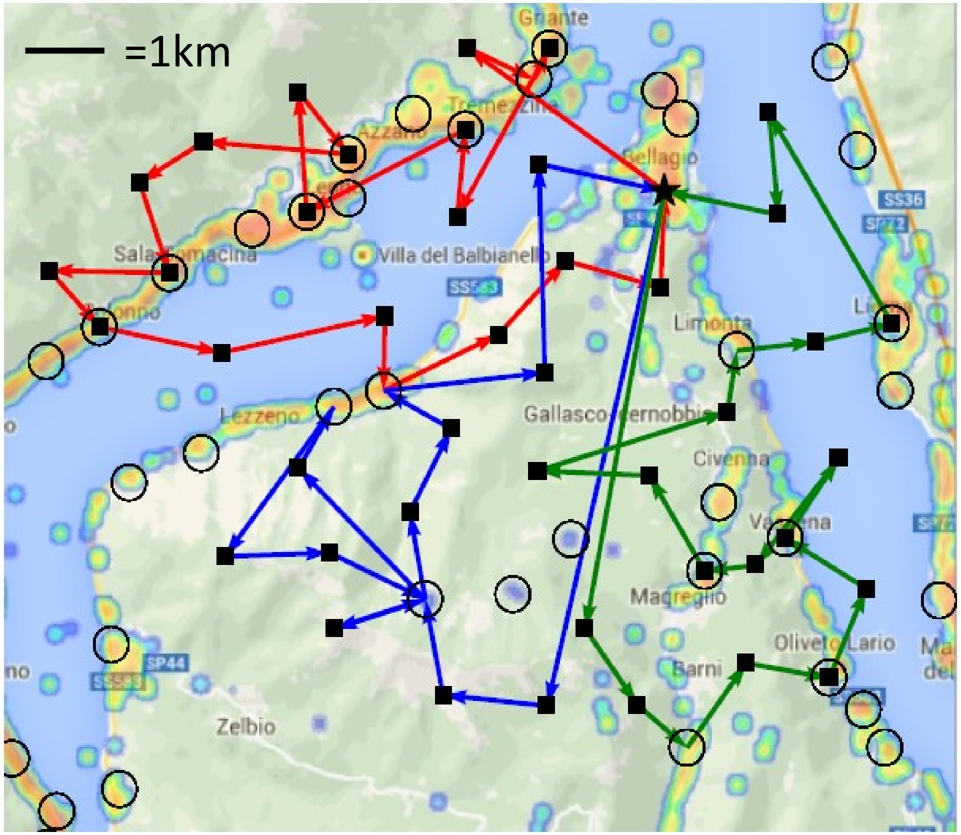
Some results

Como lake (time budget = 1h20min, 3 UAVs, speed 30 km/h)



Some results

Como lake (time budget = 1h20min, 3 UAVs, speed 30 km/h)



Avg latency:
from ≈10 min
to ≈3.5 min

Final discussion

- Agents, robots …
- Experiments …
- Projects …