Exploiting spatial locality and heterogeneity of agents for search and rescue teamwork

Maria Gini

Department of Computer Science and Engineering

University of Minnesota

(work with James Parker, Ernesto Nunes, and Julio Godoy)

Motivations and scope

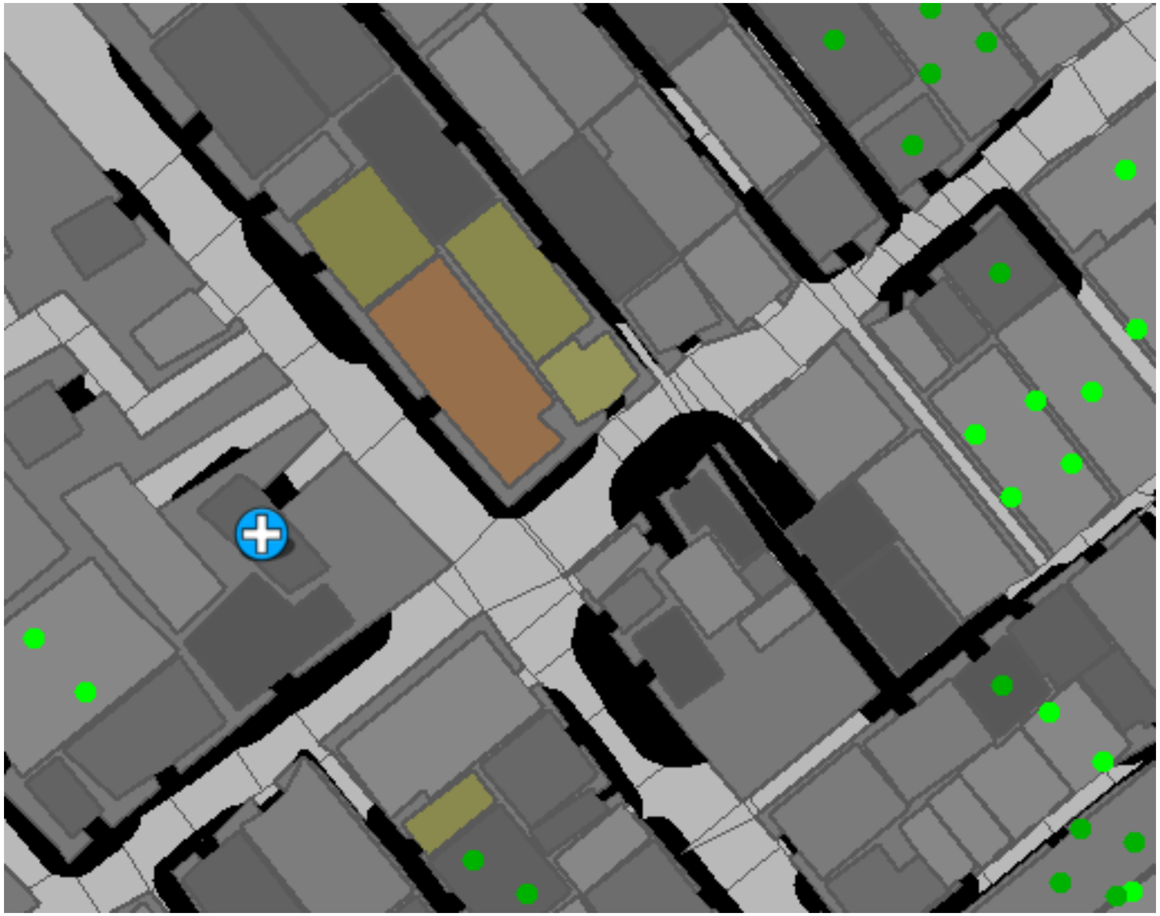
- Work done in simulation, with simulated agents that move around in a city after a disaster situation
- Focus on high-level decision procedures for the agents in the context of
  - complex environment
  - limited situational awareness
  - limited ability to communicate
  - stringent time constraints
  - heterogeneous agents
- Empirical validation in RoboCup Rescue simulation

Outline

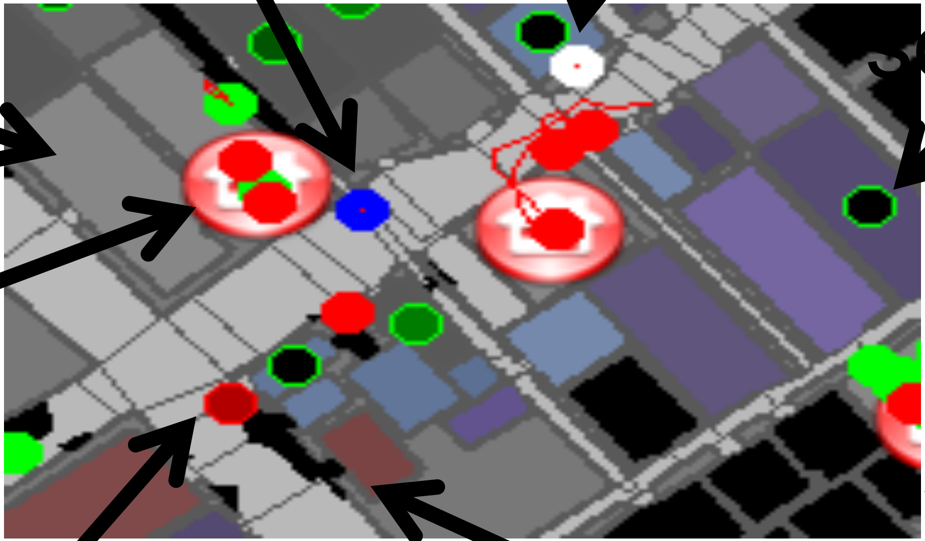
- Overview of RoboCup Rescue Simulation
- Why teamwork
- Types of teams for RoboCup Rescue
- Teamwork when the cost of tasks grows with time
- Performance evaluation
- Conclusions

RoboCup Rescue Simulator

- Open source software, fully written in Java
- Detailed graphics to visualize buildings, streets, and what each agent can see
- Ability to import real maps into the simulator



RoboCup Rescue Simulation



Blue dots = police

White dots = ambulance

Non-burning building

Unfortunate soul

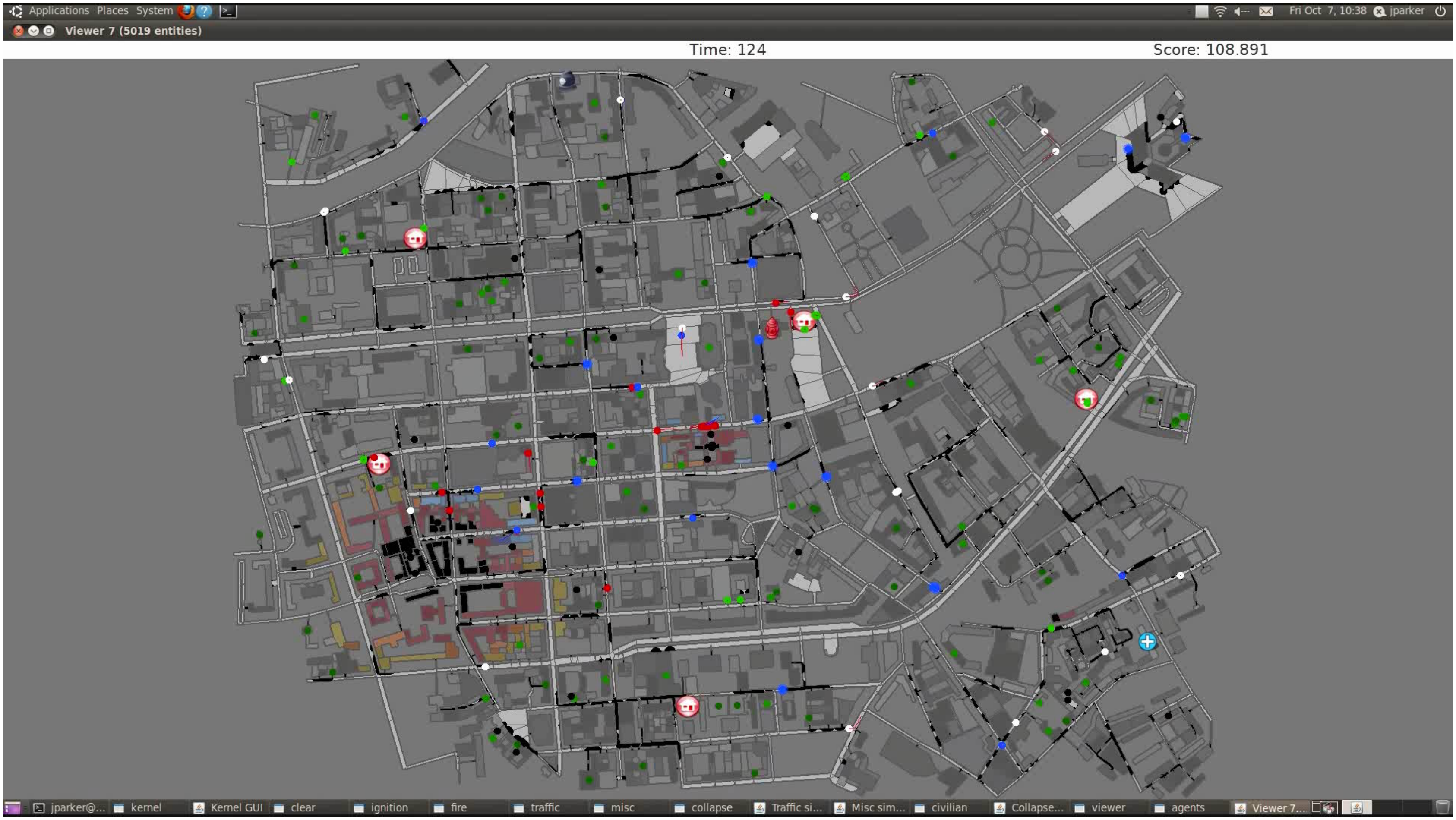
“Refuge”

Burning building

Red dots = fire trucks

Burned building

Berlin map



Viewer 7 (5019 entities)

Time: 124

Score: 108.891

Why RoboCup Rescue Simulator?

- Complex environments (real maps of real cities)
- Lack of information. Agents know the map before the disaster but do not know what roads are impassable, what buildings are collapsed, where are fires, etc
- Limited communications. Limited number of channels and number of bytes. A percentage of the message do not get delivered
- Heterogeneous agents (police, ambulances, and fire trucks) with different types of tasks

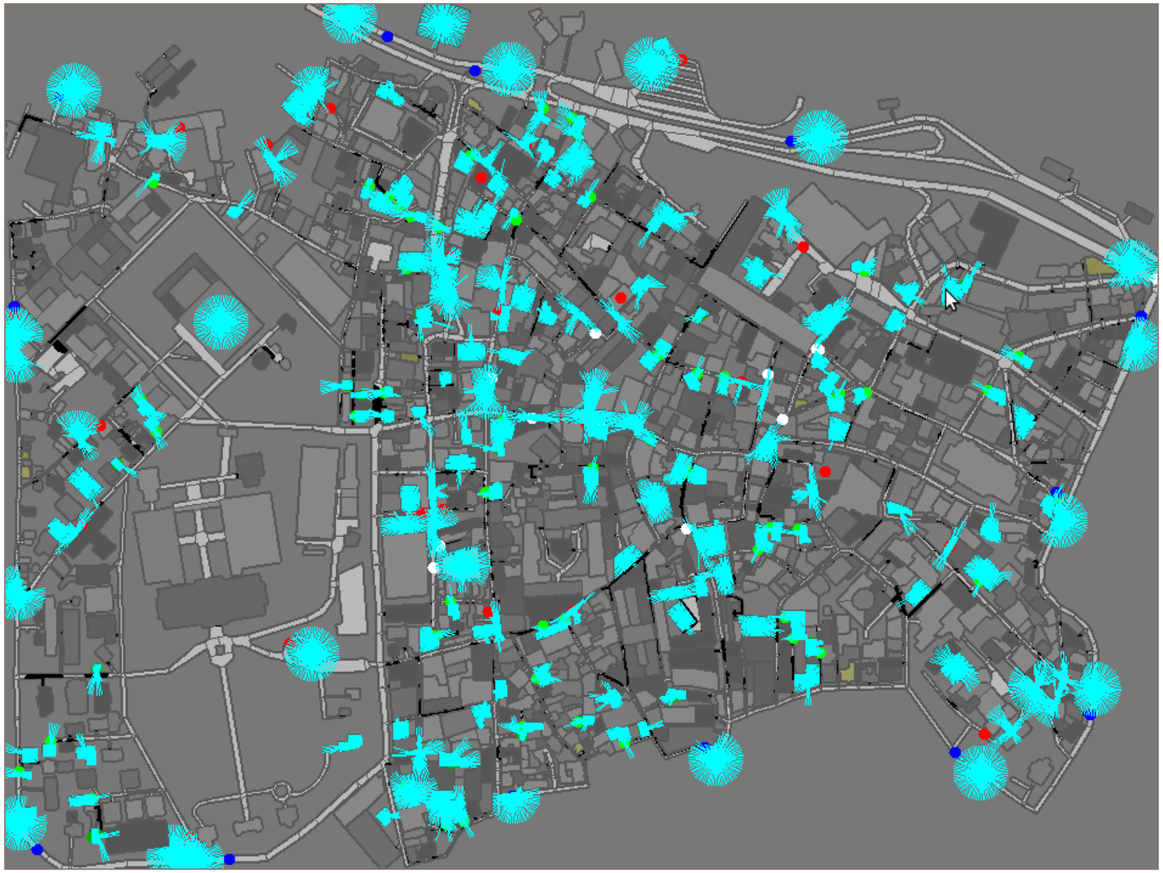
Teamwork

1. Why teamwork? Does it improve performance?
   - Exploit synergies among team members
   - Improved ability to adapt to unexpected situations
   - Robustness to individual failures
   - Structured way to share information
2. What do agents needs to do to be team members?
   - Methods for distributing tasks to agents in the team
   - Methods for coordinating the operations
3. How to create teams? Does the team structure matter?
   - We will look only at fully autonomous agents but other structures could be used (e.g., humans in the loop)

Share information among team members

Each agent has limited visibility, but when their areas of visibility are combined the team has a more precise situation assessment

This requires the ability to communicate and share information

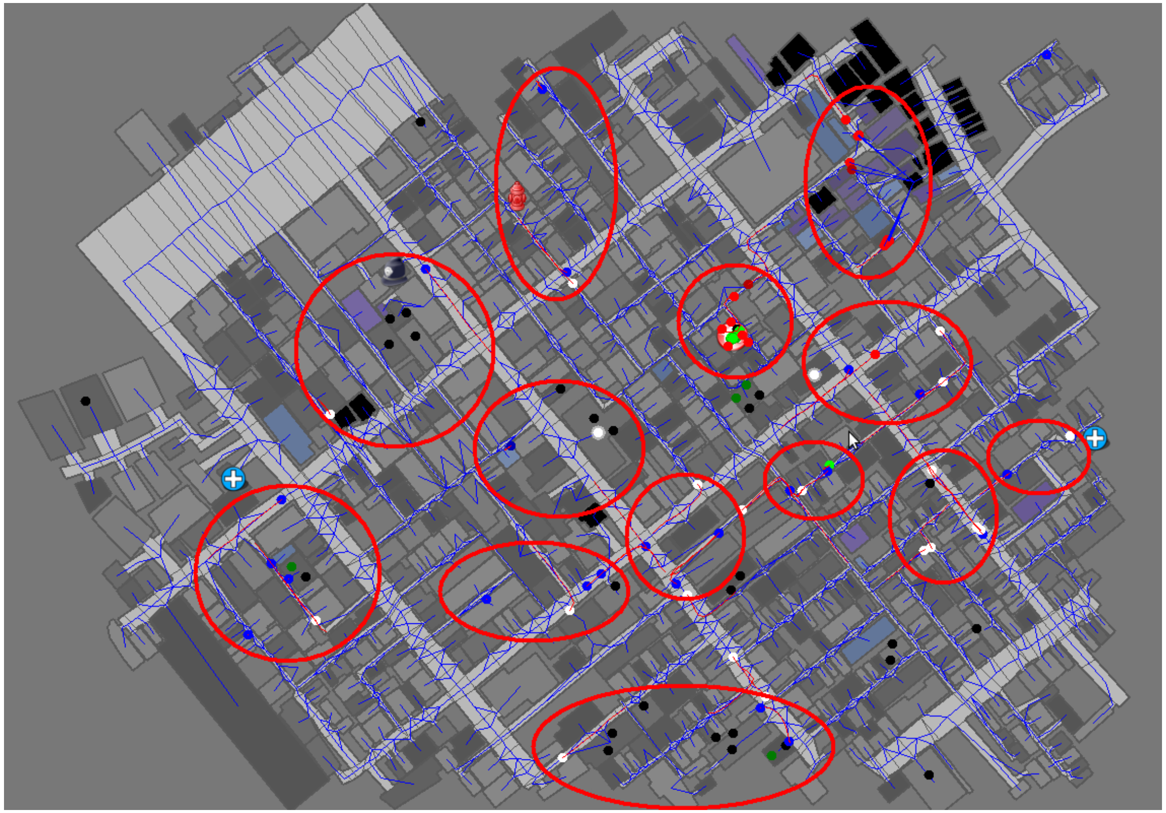


Team configurations

1. No teamwork (“base” configuration)
   - Baseline for comparisons
   - Agents share information
   - No cooperation (greedily select tasks)
2. Static teams (“static” configuration)
   - Teams formed at start
   - Agents must operate in vicinity of team
3. Dynamic teams (“dynamic” configuration)
   - Agents may switch teams if it increases utility
4. Split teams (“split” configuration)
   - Ambulance and Fire trucks in separate teams

Teamwork for tasks with locality

- ❑ Partitioning can be used for creating teams
- ❑ Cluster agents close by and assign areas to teams



Value of tasks by agent type

Police agent target *t* valuation:

$$Val_t = \frac{100 \times I_t}{D_t}$$

D_t is distance from agent to t

I_t is 1 if road t is completely blocked

otherwise rapidly goes to 0

Fire agent target *t* valuation:

$$Val_t = \frac{100}{D_t \times F_t}$$

F_t is fieriness of building t (higher = hotter)

Ambulance agent target *t* valuation:

$$Val_k = \frac{100}{D_k} - B_k.$$

B_t is depth buried (higher = deeper)

When no targets exist, agents do random search

Static teams

Teams have an area to cover. Coordinates of the team center containing agents i are:

$$\bar{x} = \frac{\sum_i x_i}{\sum_i 1}, \bar{y} = \frac{\sum_i y_i}{\sum_i 1}$$

To reduce the value of targets near the area boundary we modify the target *t* valuation:

$$TeamVal_t = Val_t * P(TD_t/TR)$$

P(t) is a Sigmoid function

TD_t is distance from team center to t

TR is a “team radius” constant

Dynamic teams

Agent utility

$$U(t)_a = \begin{cases} U(t-1)_a + R_d & \text{if the agent is doing a task,} \\ U(t-1)_a + R_m & \text{if the agent is moving to a task,} \\ U(t-1)_a + R_s & \text{if the agent is searching for a task.} \end{cases}$$

with Rd=4, Rm=1, and Rs=2

Compute the expected utility gain of moving one agent from team m*j* to team m*i* :

$$Gain(t)_{m_i,m_j,p} = \bar{U}(t)_{m_i,p} \times (W - TT(m_i,m_j)) - \bar{U}(t)_{m_j,p} \times W$$

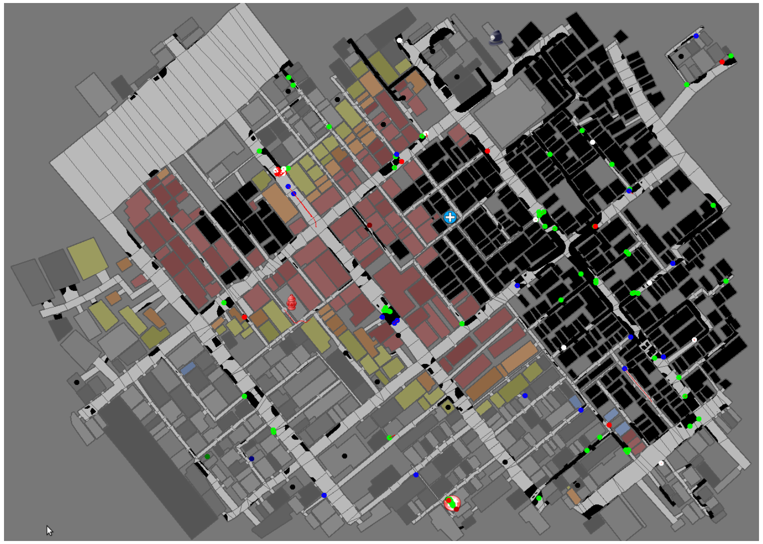
where W = time window constant and TT(mi,mj) = time to travel from team mj to mi

Split teams

Domain knowledge suggests not to have fire trucks and ambulances in the same team, because they rarely work together. We use hierarchical clustering to create teams of

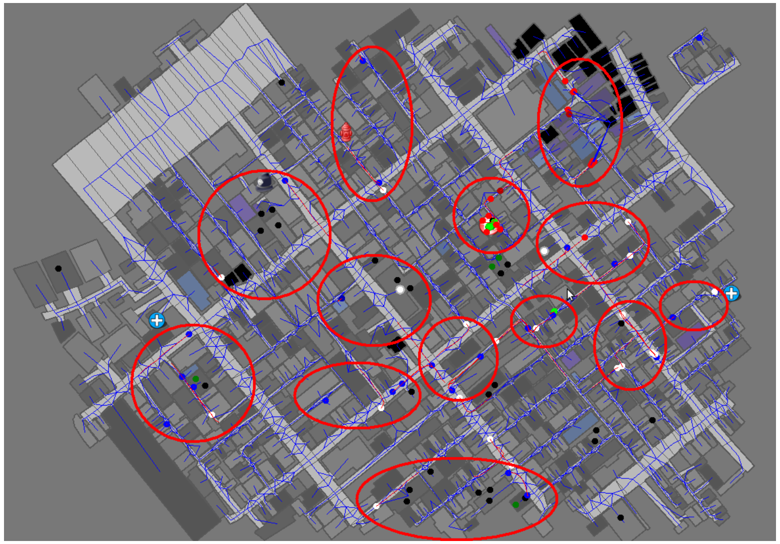
- Ambulances and police
- Fire trucks and police

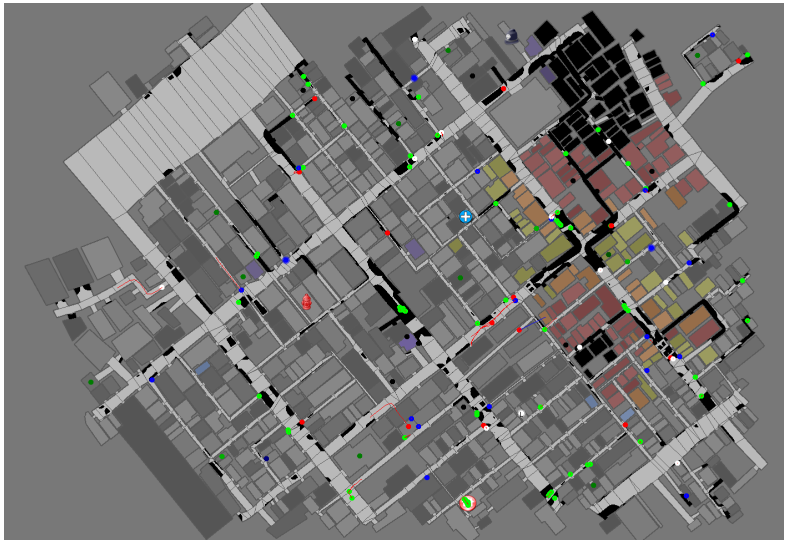
Teamwork affects performance



Example: In RoboCup Rescue Simulation without using teams a large part of the city is burned.

With teams of close-by agents there is much less fire damage.

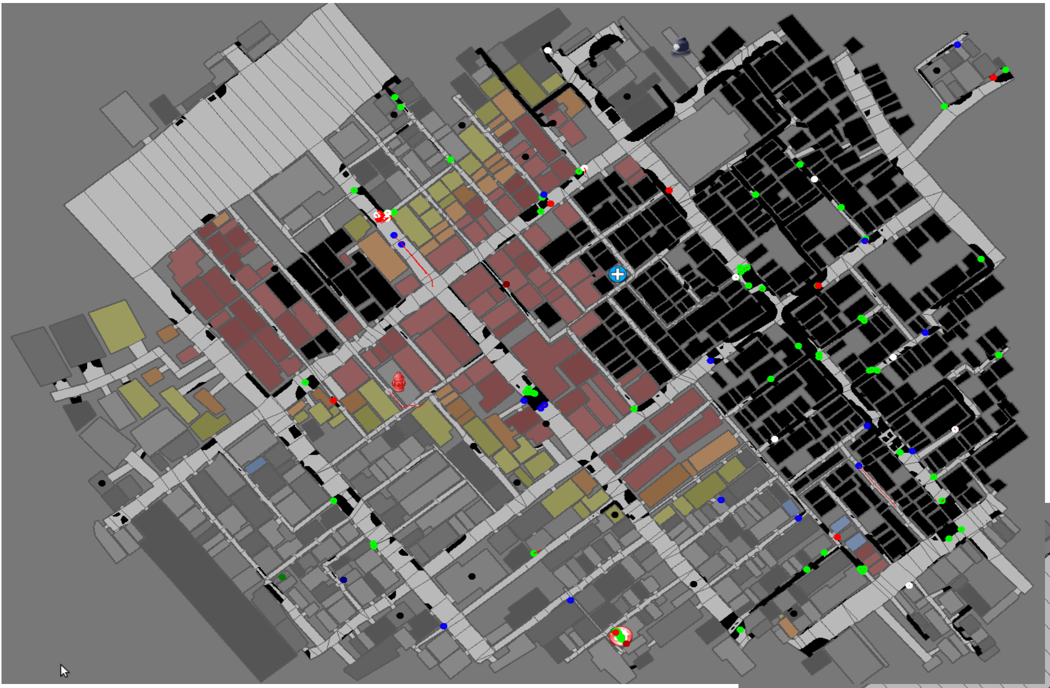




Base agents in Kobe map

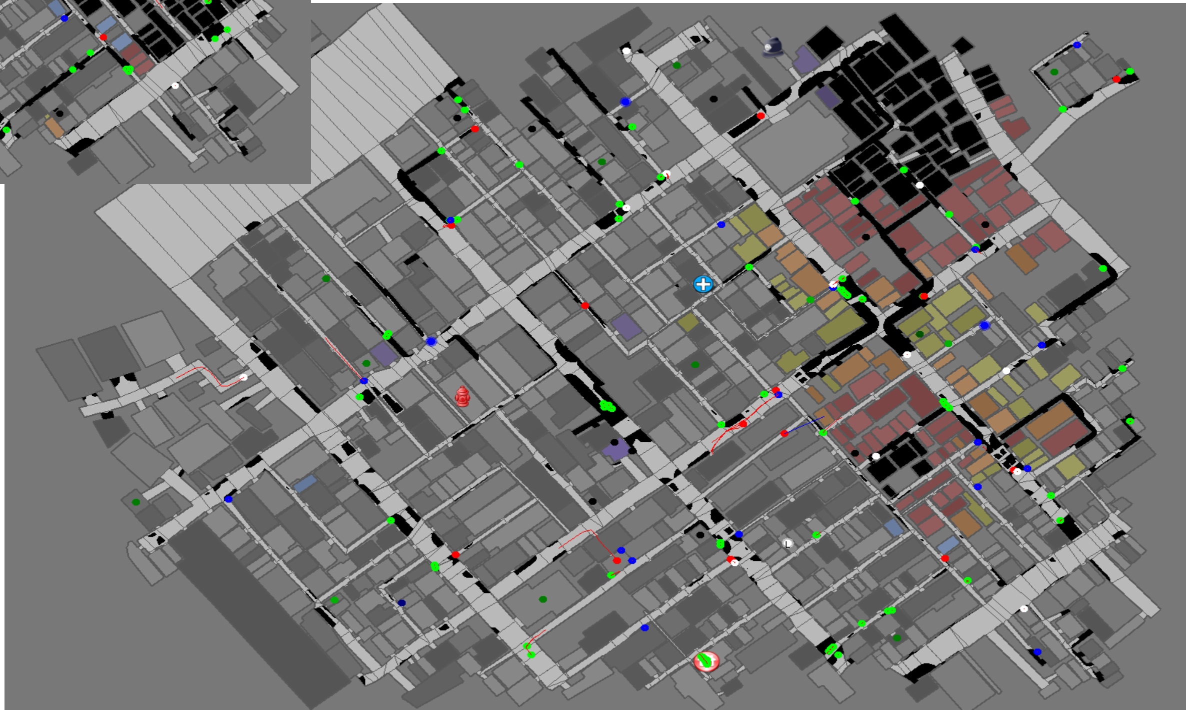
Split teams in Kobe map

Results - Kobe

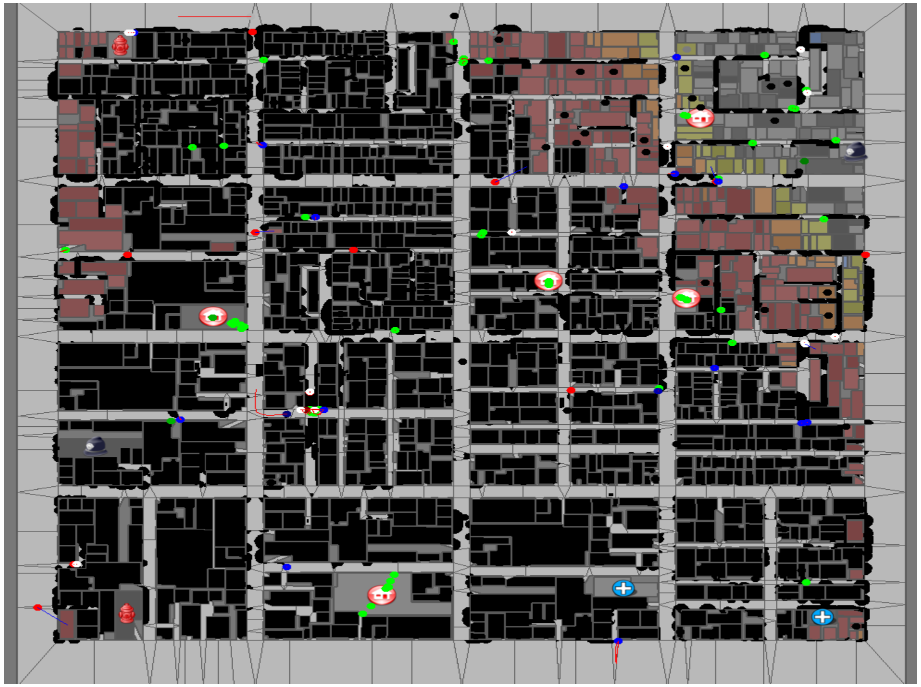


Base

Split

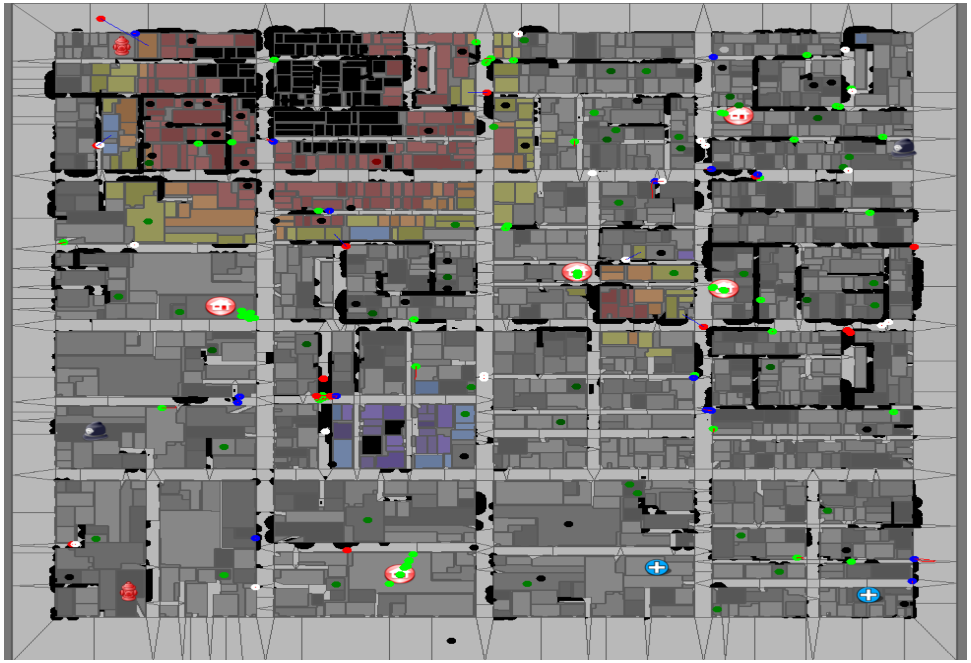


Results in VC map



Base

Split



Forming Long Term Teams to Exploit Synergies among Heterogeneous Agents, James Parker, Ernesto Nunes, Julio Godoy, Maria Gini , Tech Rep 12-016, Dept CSE, Univ of Minnesota

Competition scores in RoboCup

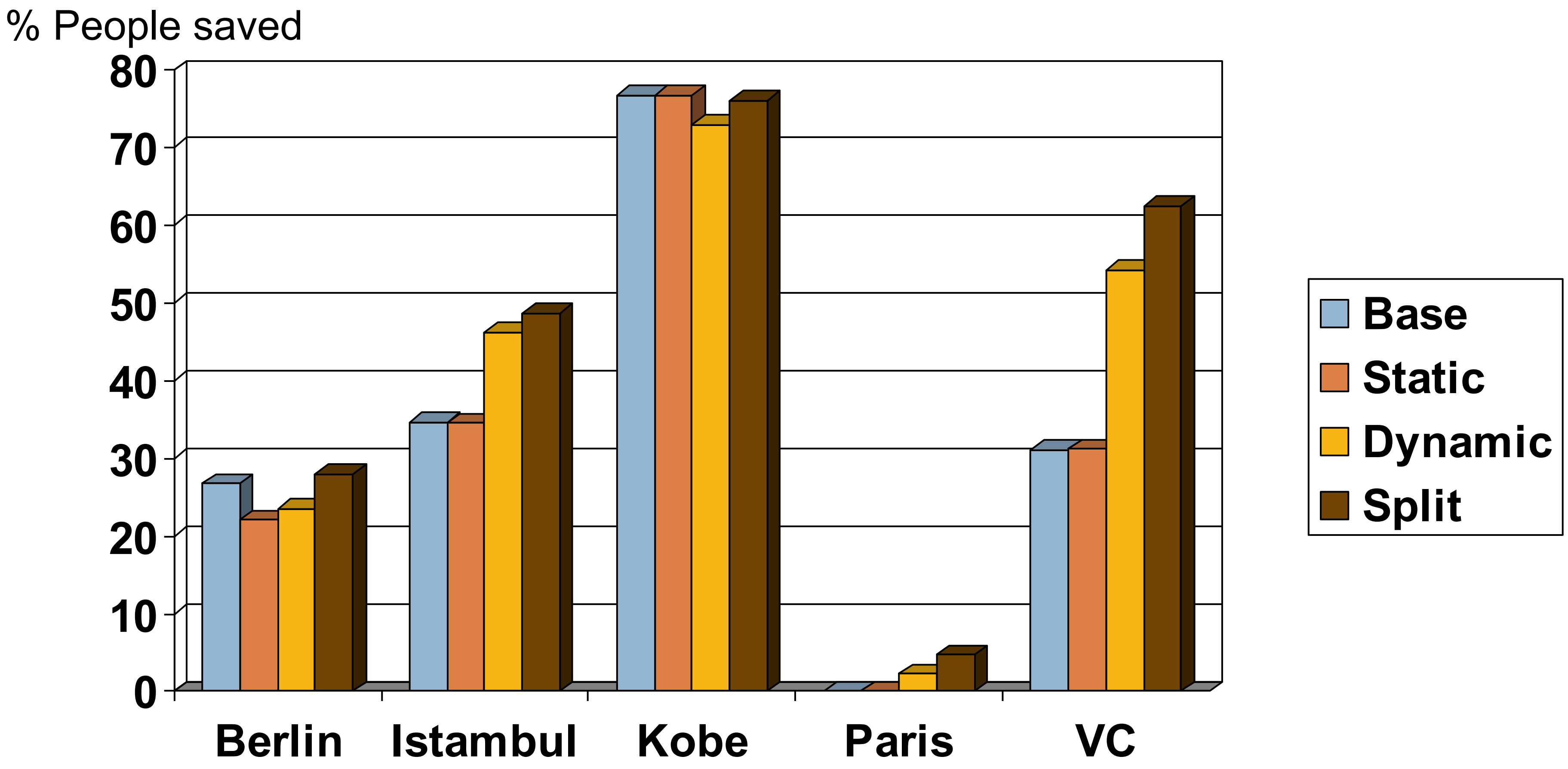
- Score formula used in RoboCup

$$(CA+CH)\times\sqrt{(BD)}$$

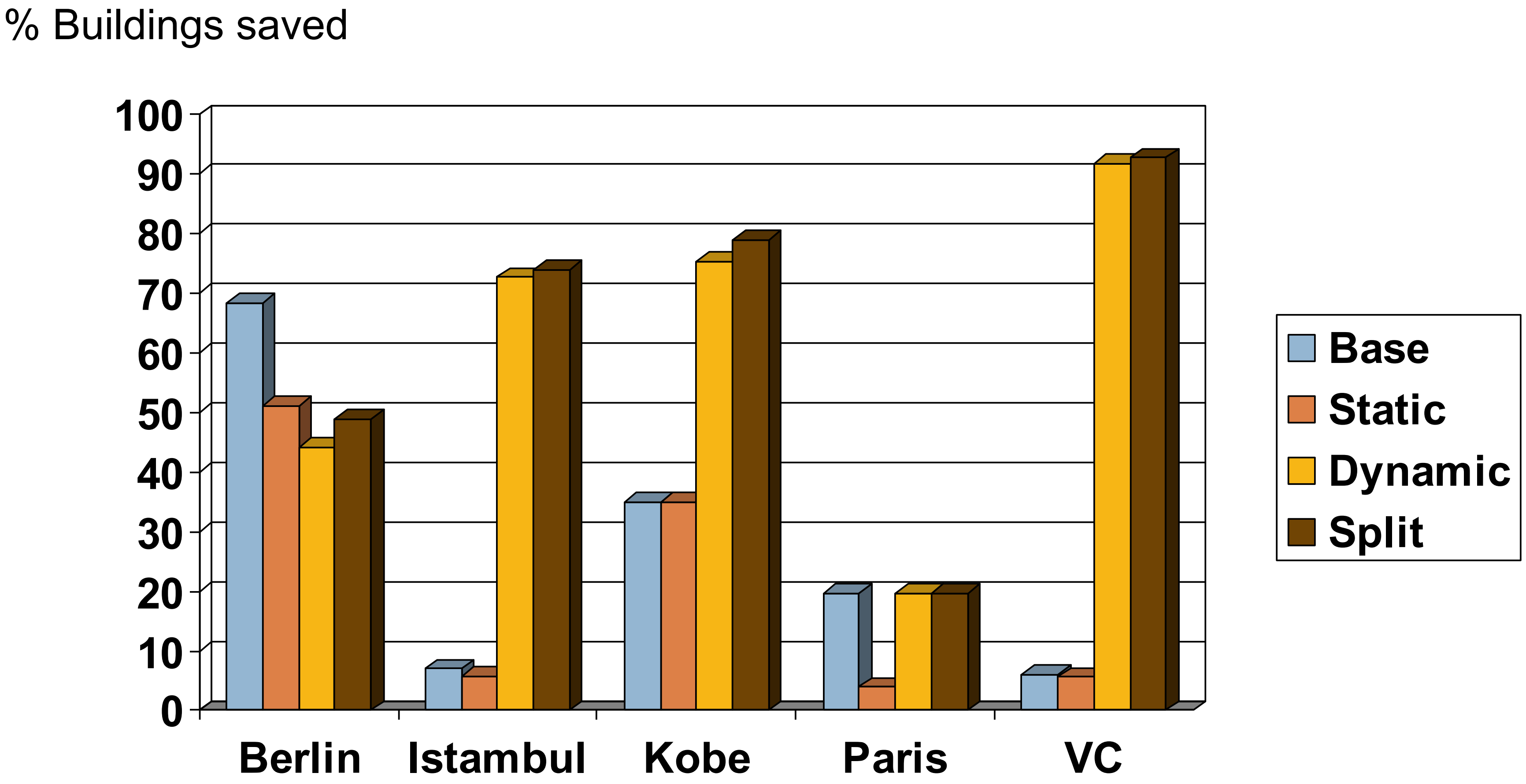
where CA is number of civilians alive, CH is the average percent of civilian health, and BD is the average percent of building health

- Need to both rescue civilians and extinguish fires well

Performance Summary: people saved

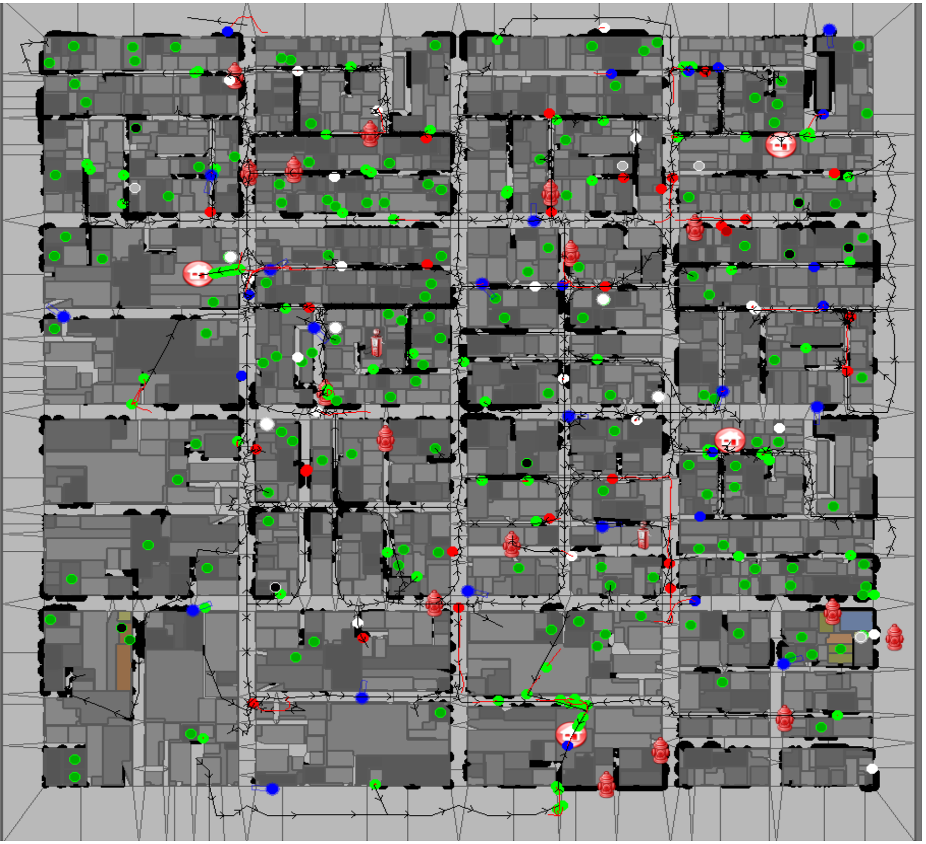


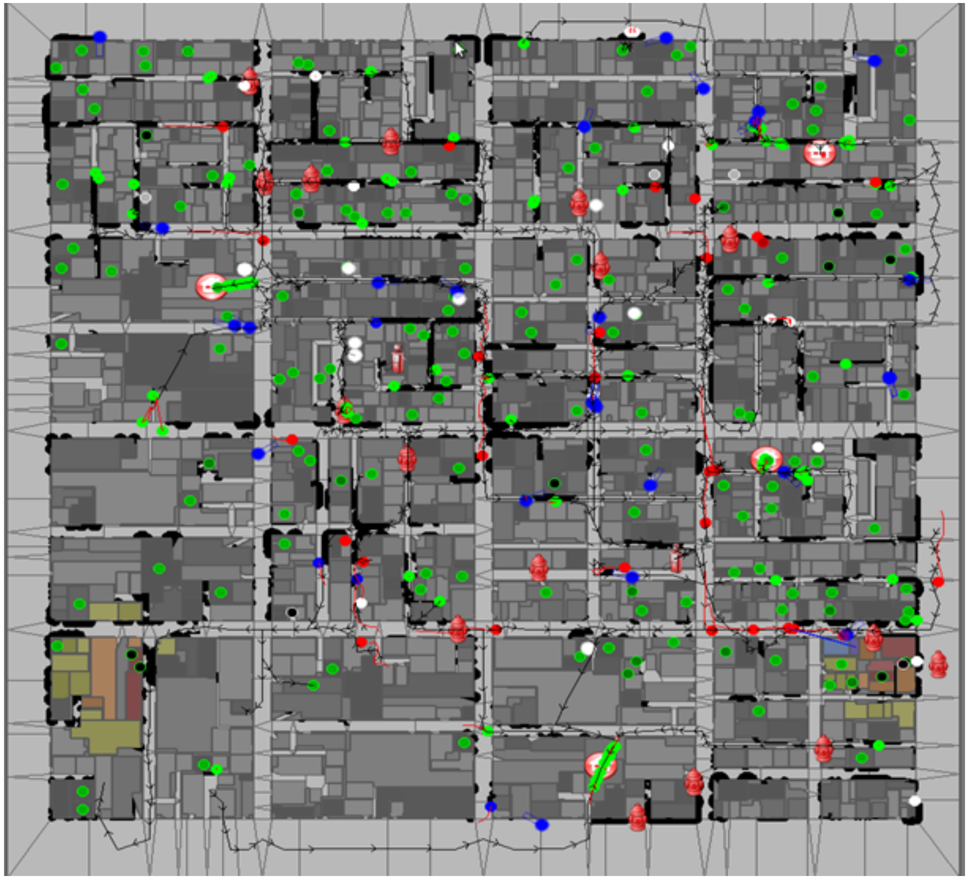
Performance Summary: buildings saved



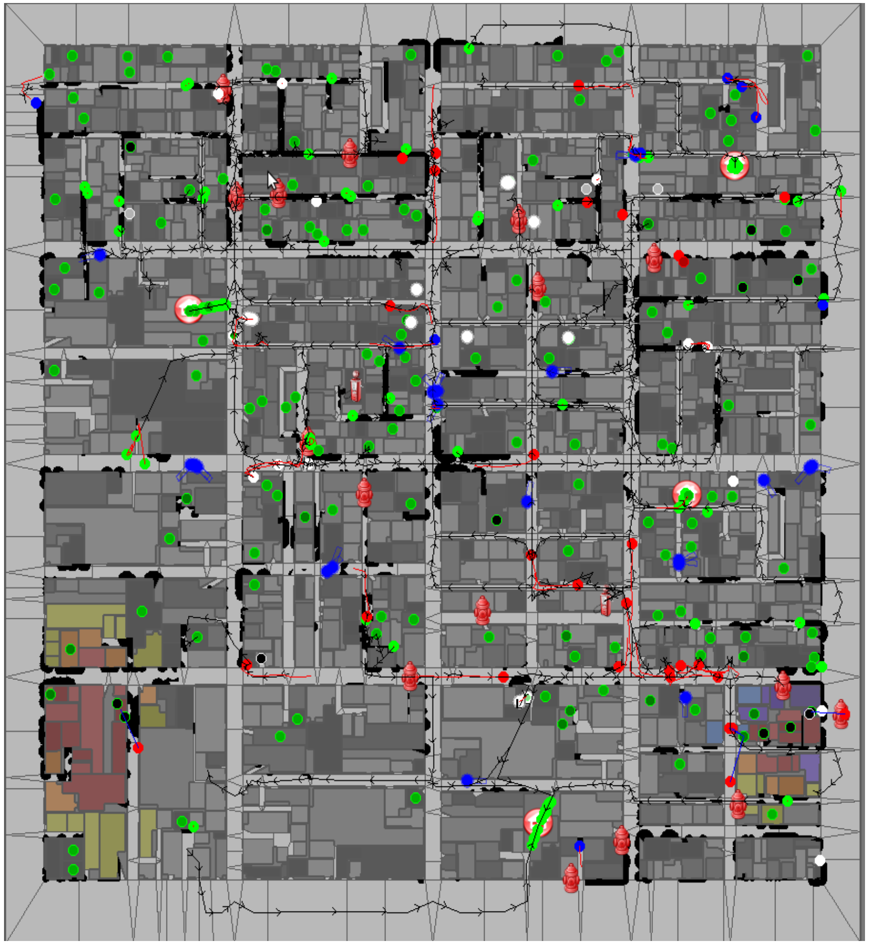
Spread of fires in VC2

A fire starts in the bottom left corner and one in the rightmost bottom corner

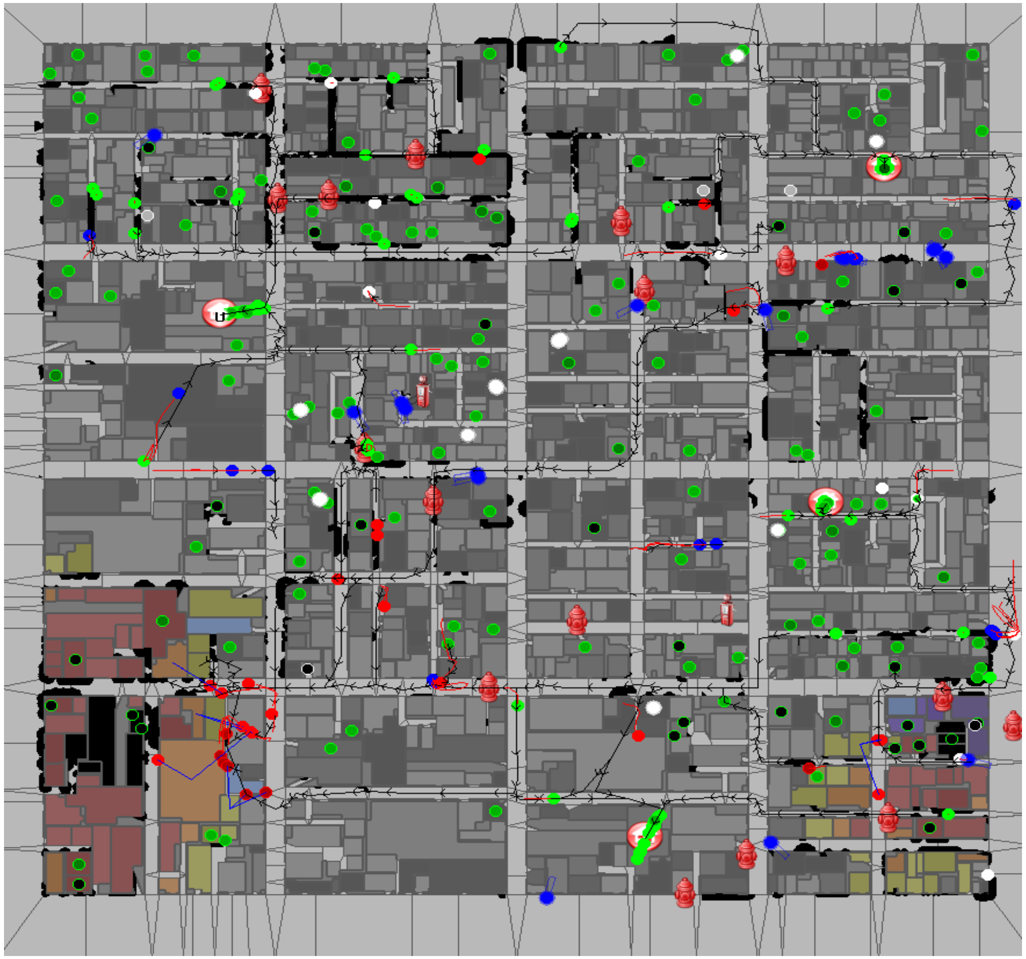




VC2 (continued)



The fire in the bottom left corner has grown out of bounds by the time fire trucks arrive



How to allocate tasks when their cost changes with time?

Fires grow with time and become harder to contain
Animal stampede become impossible to stop

Tasks with cost that grows with time

- The cost of each task bi is given by

$$f_i^{t+1} = f_i^t + \Delta f_i^t \qquad \Delta f_i^t = h_i(f_i^t) - w \times |n_i^t|$$

- As solution is found at time ts if and only if for all bi

$$f_i^{t_s} = 0.$$

$$\sum_{b_i \in B} \left(f_i^0 + \sum_{t < t_s} \left(h_i(f_i^t) - w \times |n_i^t| \right) \right) = 0$$

Latest Finishing First (LFF)

Create an initial stable assignment of agents to tasks and try to maximize the work done by the agents

$$\sum_{b_i \in B} f_i^0 + \sum_{b_i \in B} \sum_{t<t_s} h_i(f_i^t) - \bar{p} \times t_s \times w \times |A| = 0$$

We do this by assigning agents to the task which finishes last, since this will reduce the value of the term that grows with time.
First we assign the tasks that never finish, selecting for them the closest agent still free. When all those tasks are assigned we find the task with the largest finish time and assign the closest agent and continue until all agents are assigned. Larger tasks (i.e. with larger finish times) will get more agents.

The algorithm terminates when all the agents are assigned. If the task growth model is accurate and the travel time is the same the allocation is optimal.

Real-Time Latest Finishing First (RT-LFF)

Since the estimates for tasks growth might not be correct and new tasks can appear, after the initial allocation is done using LFF, the algorithm RT-LFF adjusts the allocation checking each pair of tasks to see if moving an agent from one to the other can improve the allocation. If the completion time for task i with one fewer agent is smaller than the completion time for task j them one agent is moved from task i to task j. The order in which the pairs are checked is critical. New tasks are considered first paired with their closest task because the travel time of the agent transferred, if any, will be the smallest.

Modeling fires in RoboCup

- Rate of growth of fire (g) is proportional to the number of buildings on fire (x)

  (a) $\frac{\delta x}{\delta t} = g \times x$ (b) $x = C \times e^{g \times t}$

- The rate of growth is reduced by the number of agents (n) working on it times the work each does (w)

  $$\frac{\delta x}{\delta t} = g \times x - n \times w$$

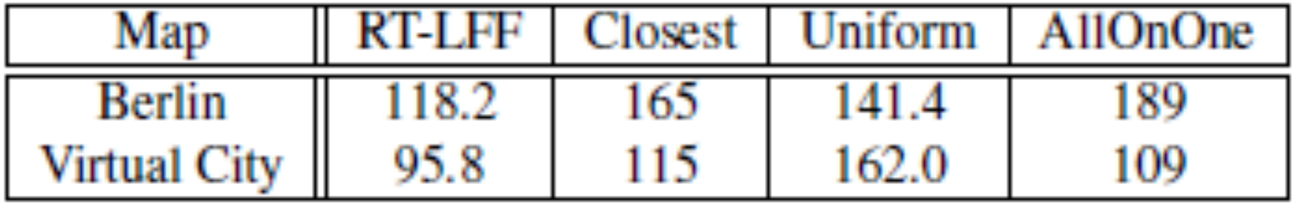
- From this we compute the number of buildings on fire

  $$x = \frac{n \times w}{g} + C \times e^{g \times t}$$

- Solving for x=0 produces the completion time (ct) when the last fire will be extinguished $ct = (\ln \frac{n \times w}{g \times -C})/g$

Experimental results

- Compared RT-LFF with
  - sending an agent to the closet fire in need,
  - uniform distribution of agents to tasks,
  - assigning all agents to one task and move them when the task is complete



Map	RT-LFF	Closest	Uniform	AllOnOne
Berlin	118.2	165	141.4	189
Virtual City	95.8	115	162.0	109

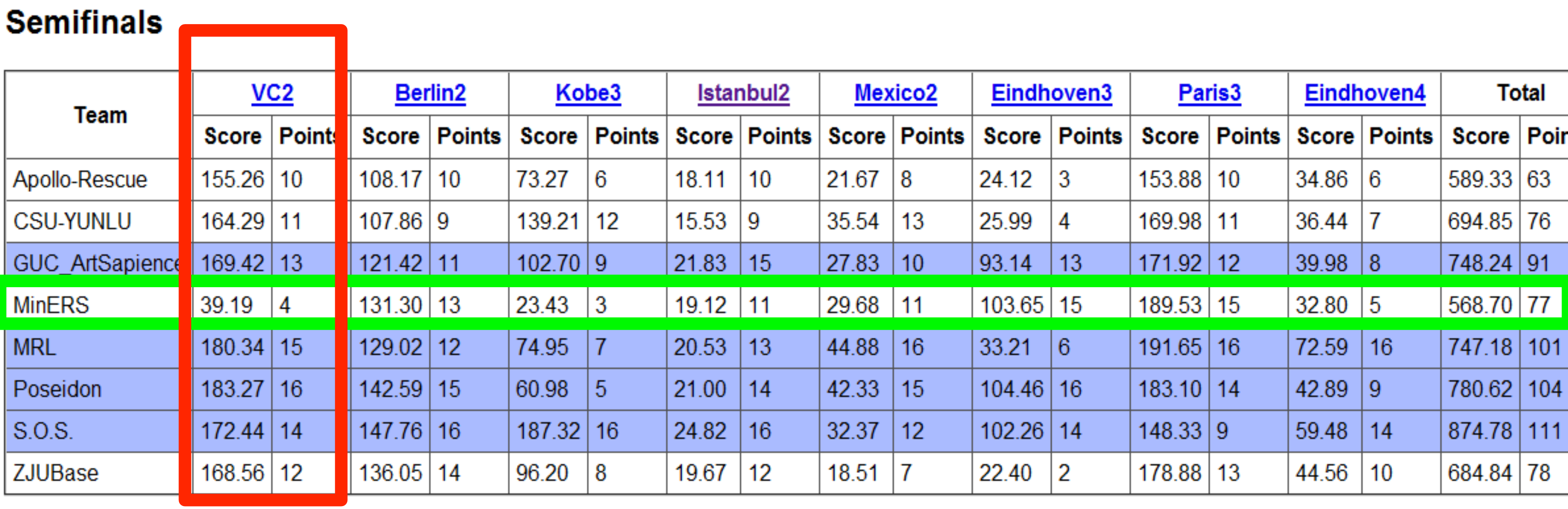
Observations on RoboCup 2013 results

- The distribution of scores for each map tends to have two clusters
  - The agents that extinguish fires well
  - The agents that cannot control the fires
-

2013 RoboCup MinERS rankings:

Preliminary - 4th

Semi-finals - 6th

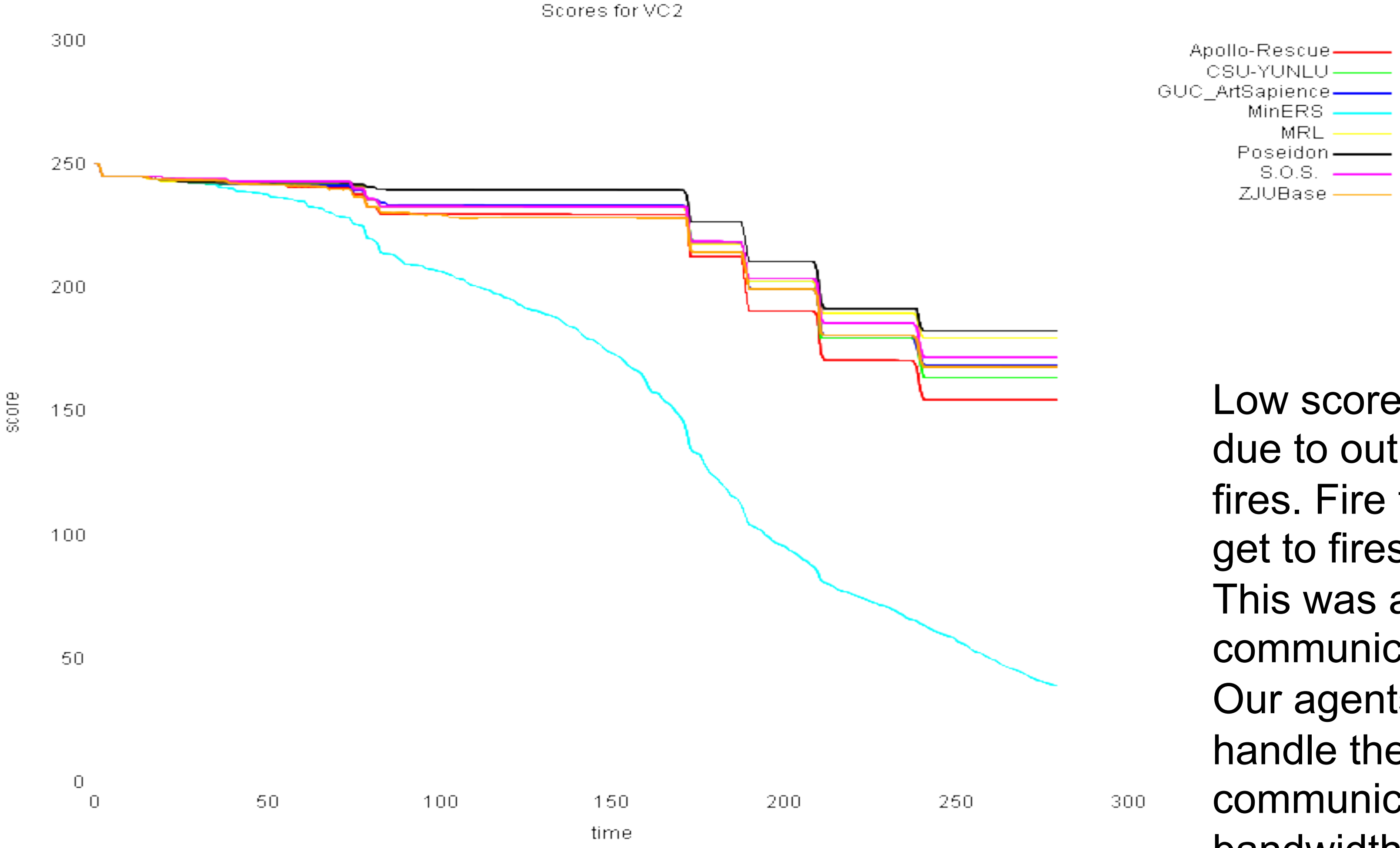


Semifinals

Team	VC2		Berlin2		Kobe3		Istanbul2		Mexico2		Eindhoven3		Paris3		Eindhoven4		Total	
	Score	Points	Score	Points	Score	Points	Score	Points	Score	Points	Score	Points	Score	Points	Score	Points	Score	Poir
Apollo-Rescue	155.26	10	108.17	10	73.27	6	18.11	10	21.67	8	24.12	3	153.88	10	34.86	6	589.33	63
CSU-YUNLU	164.29	11	107.86	9	139.21	12	15.53	9	35.54	13	25.99	4	169.98	11	36.44	7	694.85	76
GUC_ArtSapience	169.42	13	121.42	11	102.70	9	21.83	15	27.83	10	93.14	13	171.92	12	39.98	8	748.24	91
MinERS	39.19	4	131.30	13	23.43	3	19.12	11	29.68	11	103.65	15	189.53	15	32.80	5	568.70	77
MRL	180.34	15	129.02	12	74.95	7	20.53	13	44.88	16	33.21	6	191.65	16	72.59	16	747.18	101
Poseidon	183.27	16	142.59	15	60.98	5	21.00	14	42.33	15	104.46	16	183.10	14	42.89	9	780.62	104
S.O.S.	172.44	14	147.76	16	187.32	16	24.82	16	32.37	12	102.26	14	148.33	9	59.48	14	874.78	111
ZJUBase	168.56	12	136.05	14	96.20	8	19.67	12	18.51	7	22.40	2	178.88	13	44.56	10	684.84	78

Oops...

VC2 Competitor Comparison



Low score of MinERS due to out of control fires. Fire trucks do not get to fires quickly. This was a communication issue... Our agents did not handle the limited communication bandwidth in this map

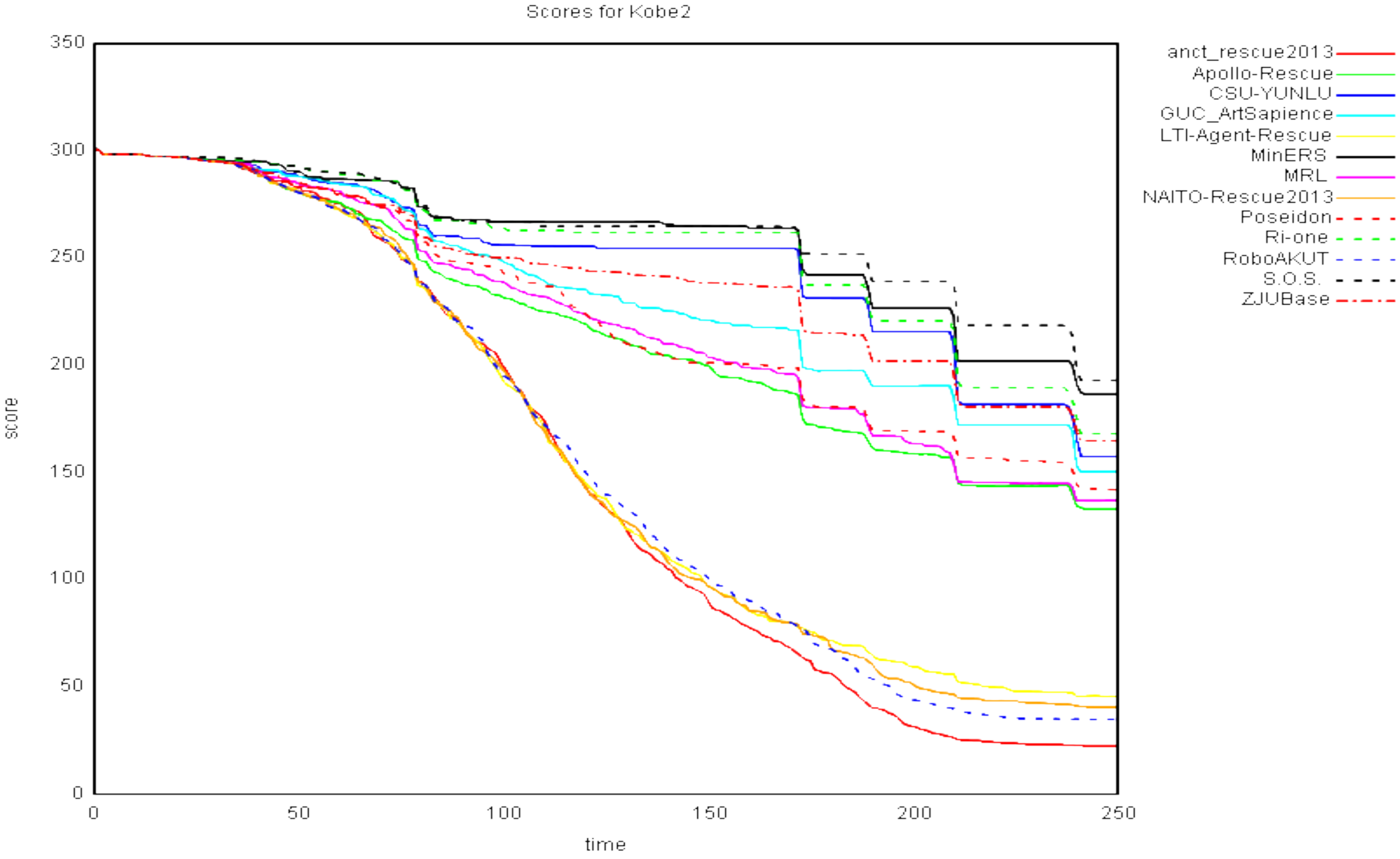
No Channels for Communication

- Our teamwork framework:
  - has low communication requirements
  - allows implicit coordination

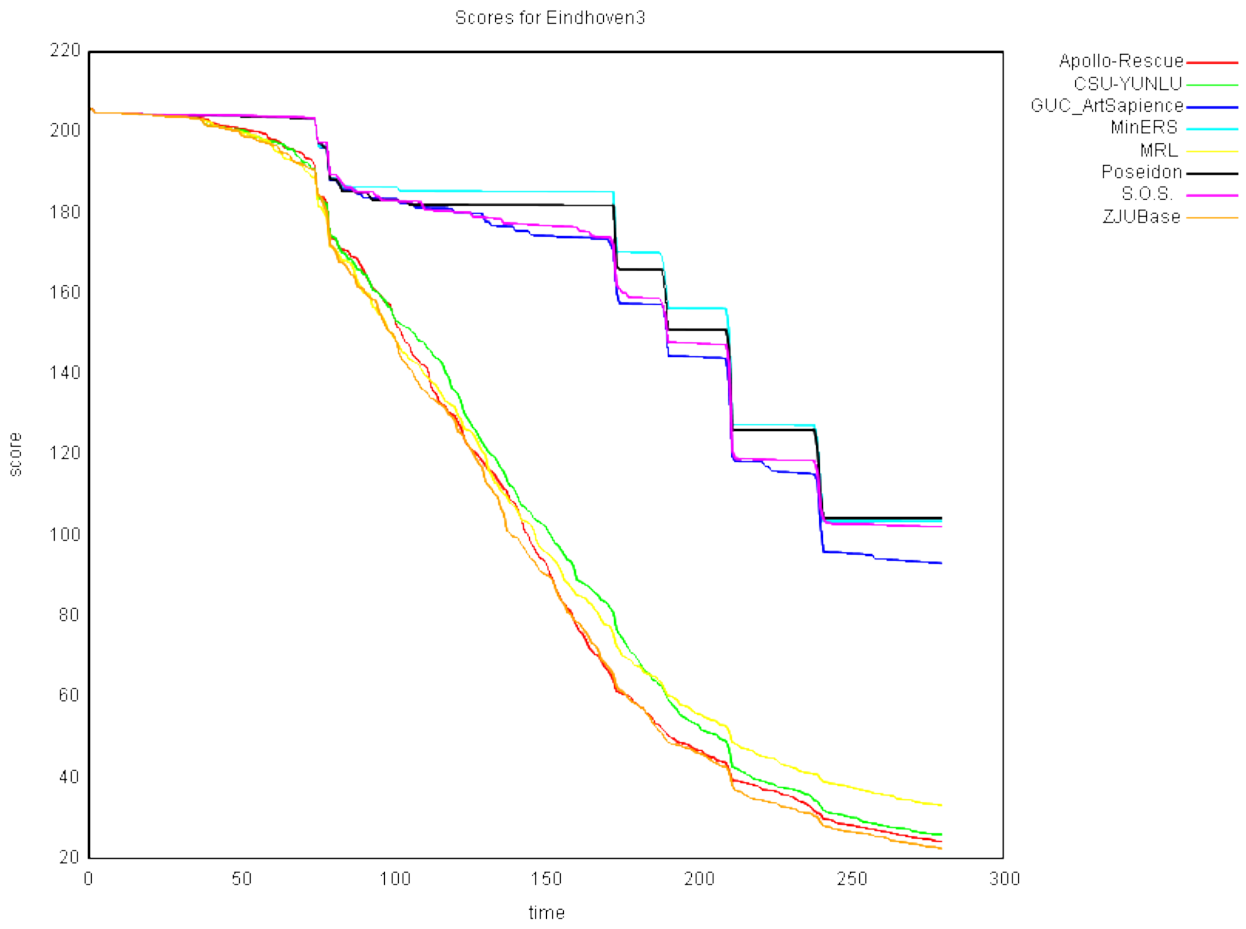
- Teams are spatially local, so an agent's voice can reach most teammates.

- On maps with only voice (no communication channel) the MinERS team does well against competitors

No Communication



Distribution of Scores



RoboCup 2013 results

Analysis of the results shows that the difference among the top teams is not statistically significant. More maps and more runs are needed to obtain statistical significance.

Conclusions

- We have shown methods for creating teams and shown that some types of team work together better than others (ambulances and police, or fire brigades and police)
- Demonstrated that partitioning agents into teams increase performance
- Studied allocation of agents to tasks with costs that grow with time and proposed algorithms
- Presented preliminary results on performance in RoboCup competition

Thanks for your attention!

For more information go to http://www.cs.umn.edu/~gini

or email to gini@cs.umn.edu

Thanks to James Parker, Ernesto Nunes, and Julio Godoy for their work