A Networking Framework for Multi-Robot Coordination

Article

Accepted version

D. Peri, G. Lo Re, M. Ortolani, A. Chella

In Recent Advances in Multi Robot Systems, 2008, pp. 1-14

Publisher: InTech

http://cdn.intechopen.com/pdfs/872/InTech-A_networking_framework_for_multi_robot_coordination.pdf

Chapter Number

A Networking Framework for Multi-Robot Coordination

Antonio Chella, Giuseppe Lo Re, Irene Macaluso, Marco Ortolani, and Daniele Peri
Dipartimento di Ingegneria Informatica, Università di Palermo
Viale delle Scienze, 90128 Palermo, Italy

1. Introduction

Autonomous robots operating in real environments need to be able to interact with a dynamic world populated with objects, people, and, in general, other agents.
The current generation of autonomous robots, such as the ASIMO robot by Honda or the QRIO by Sony, has showed impressive performances in mechanics and control of movements; moreover, recent literature reports encouraging results about the capability of such robots of representing themselves with respect to a dynamic external world, of planning future actions and of evaluating resulting situations in order to make new plans. However, when multiple robots are supposed to operate together, coordination and communication issues arise; while noteworthy results have been achieved with respect to the control of a single robot, novel issues arise when the actions of a robot influence another's behavior.
The increase in computational power available to systems nowadays makes it feasible, and even convenient, to organize them into a single distributed computing environment in order to exploit the synergy among different entities. This is especially true for robot teams, where cooperation is supposed to be the most natural scheme of operation, especially when robots are required to operate in highly constrained scenarios, such as inhospitable sites, remote sites, or indoor environments where strict constraints on intrusiveness must be respected.
In this case, computations will be inherently network-centric, and to solve the need for communication inside robot collectives, an efficient network infrastructure must be put into place; once a proper communication channel is established, multiple robots may benefit from the interaction with each other in order to achieve a common goal.
The framework presented in this paper adopts a composite networking architecture, in which a hybrid wireless network, composed by commonly available WiFi devices, and the more recently developed wireless sensor networks, operates as a whole in order both to provide a communication backbone for the robots and to extract useful information from the environment.
The ad-hoc WiFi backbone allows robots to exchange coordination information among themselves, while also carrying data measurements collected from surrounding environment, and useful for localization or mere data gathering purposes.

The proposed framework is called *RoboNet*, and extends a previously developed robotic tour guide application (Chella et al., 2007) in the context of a multi-robot application; our system allows a team of robots to enhance their perceptive capabilities through coordination obtained via a hybrid communication network; moreover, the same infrastructure allows robots to exchange information so as to coordinate their actions in order to achieve a global common goal.

The working scenario considered in this paper consists of a museum setting, where guided tours are to be automatically managed. The museum is arranged both chronologically and topographically, but the sequence of findings to be visited can be rearranged depending on user queries, making a sort of dynamic virtual labyrinth with various itineraries. Therefore, the robots are able to guide visitors both in prearranged tours and in interactive tours, built *in itinere* depending on the interaction with the visitor: robots are able to rebuild the virtual connection between findings and, consequently, the path to be followed.

This paper is organized as follows. Section 2 contains some background on multi-robot coordination, and Section 3 describes the underlying ideas and the motivation behind the proposed architecture, whose details are presented in Sections 4, 5, and 6. A realistic application scenario is described in Section 7, and finally our conclusions are drawn in Section 8.

2. Related Works

In the past years, various classifications of multi-robot systems have been proposed. Dudek, et al., for instance, have proposed a taxonomy on communication mechanism and their cost to highlight that different multi-robot systems have very different capabilities (Dudek et al., 1996). The taxonomy proposed by Dudek takes into account some criteria, such as the number of robots in the collective, the maximum distance between robots such that communication is still possible, the communication topology, the composition of the collective, and the computational model of individual robots. Some of the works presented in literature achieve coordination among robots through distributed control, as in the case of the Alliance architecture (Mataric, 1997), where a robot increases the utility measure for the task that it is currently accomplishing while it decreases it for all other tasks; each robot then observes the behavior of its team-mates and selects the fastest achievable task. On the other side, the MARTHA project (Alami et al., 1998) assumes a centralized control to coordinate a team of autonomous robots for transport application in structured environment.

Parker in (Parker, 2003) presented a review of the main topic areas of research regarding multi-robot systems: biological inspired robot teams, communication, architectures and task planning, localization and mapping, object transport and manipulation, motion coordination, reconfigurable robotics, learning.

A large amount of research has been dedicated to the issue of communication in multi-robot systems. Several studies have been conducted to assess the benefits provided by communication on the performance of a robot team. Balch and Arkin conducted experiments with robots equipped with LED indicators signaling the state they were in (Balch & Arkin, 1994). The results indicated that communication considerably improves system performance. Simple communication strategies are preferable, because more complex approaches do not significantly improve results.

Mataric (Mataric, 1998) used communication to share data between robots in order to compensate for the limitations of direct sensory modalities. The proposed networking framework allows the robots to enhance their perceptive capabilities by sharing the knowledge they own or the information provided by the sensor network. The features of such networks have been exploited, for instance, to allow each robot to detect people being beyond the range of robot sensors. This information is used to approach visitors and offer a guided tour of the museum.

Some of the works have focused on the issues related to fault-tolerance in multi-robot communication. Winfield developed ad-hoc wireless networking for collecting sensory data from a team of mobile robots (Winfield, 2000); the author also addressed the case where the ad-hoc wireless network is not fully connected, and rather it is partitioned into smaller sub-nets.

The scenario we are considering in the present work, however, involves only indoor communications, and an area that spans a building, so that full connection for the WiFi LAN can be reasonably assumed. Besides acting as a communication infrastructure for the robot team, the wireless LAN will also offer support for knowledge sharing; in particular, it will act as a backbone for exchanging information derived from locally deployed wireless sensor networks (WSNs).

Recently, this technology has been employed to tackle the task of closely monitoring and localizing moving objects in a structured environment (Akyildiz et al., 2002). Such networks are typically used for pervasive environmental monitoring through measurement of characteristic quantities, but each sensor node also has limited processing capabilities that may be exploited in order to carry on preliminary operations on raw data.

These features have been exploited by designing sensor nodes equipped with specialized hardware that enables them to compute a sufficiently accurate estimate of distances; such nodes have been successfully employed in the design of an indoor localization system (Priyantha et al., 2000). Such system may be employed as support for robots navigation and localization.

The synergy between wireless sensor networks and robotics has been analyzed for instance in (Moore et al., 2004), where the authors build a network of mobile sensors that can be controlled in order to collect samples of a distribution of interest, and also in (McMickell et al., 2003; Bergbreiter & Pister, 2003), where some general design, cost and scalability issues are discussed.

3. The RoboNet Framework

The design of the proposed framework has been mainly motivated by the experience developed in the context of the experiments conducted at the Archaeological Museum of Agrigento, Italy where findings from the close "Valley of the Temples", one of the UNESCO World Heritage Sites, are collected.

The purpose of the framework is the coordination of a group of robots moving in a structured indoor environment in order to manage automatically guided museum tours. Museum managers would also like to be able to provide virtual visits: tourists might thus be able to browse the exposed findings, for instance during off-peak hours, through web-based interface and could partially customize their visit by controlling the robots' actions. The quality of real visits, on the other hand, could be improved by careful planning, but this requires collection information and studying the tourist flows; moreover, it would be desirable to gather information also for surveillance purposes, so these are further *desiderata* for our framework. Finally, very tight requirements were posed by the museum board of directors regarding severe limitations on the deployment of any intrusive hardware devices on the museum premises.

The RoboNet robot team may thus be regarded as a community of connected entities with the possibility of exchanging messages with each other, and of cooperating toward achieving a shared goal in order to find the solution to a common problem. Besides the benefits that are expected to emerge from cooperation, a major difficulty arises from the fact that all involved entities must achieve efficient coordination, while acting independently and autonomously from each other; moreover, an additional difficulty is represented by the need of providing fault tolerance to the whole system.
In our architecture, we assume that the role of coordinator is not statically assigned to a single external entity; rather, at a given moment in time, one robot will take up the role of team leader, with the possibility of releasing it in favor of one of the other components of the team.
Fig. 1 shows the main components of the RoboNet framework, and represents the 3-tier architecture that has been devised to separate the main modules into functionally correlated layers. The lowest layer (the Communication Layer) is meant to provide basic connectivity by means of two different network technologies. A wireless sensor network is deployed in each room and will assist a close-by robot with self-localization tasks, by using a combination of radio frequency and ultra-sound signals, as will be explained later. Such WSNs do not form a connected network, and localization signals will not propagate across neighboring environments. Furthermore, additional wireless sensor nodes are present in the rooms; they are carried by visitors in order to provide a robot with proximity measurements, through computations performed based on RSSI signals. Those sensors will only need to be connected to a close-by robot that will thus compute a rough estimate of the distance of people the are supposedly following the tour it is guiding.
On the other hand, the WiFi backbone will provide connection among all robots, and will be used for exchanging messages related to coordination, or limited knowledge sharing.

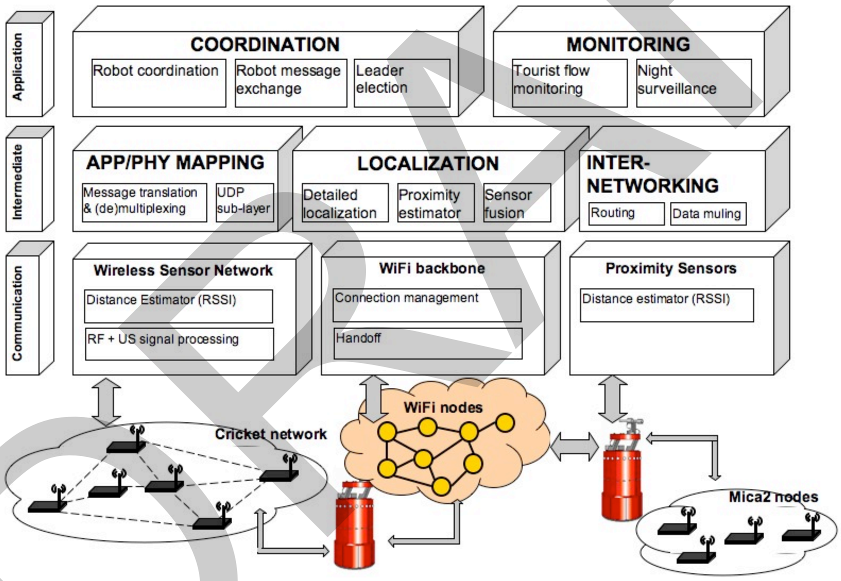


Fig. 1: The RoboNet architecture.

One layer up in the proposed architecture, the Intermediate Layer is where data previously collected are aggregated to extract information useful for localization or event detection purposes, such as signaling people abandoning the visit. This layer will also deal with specific networking issues, such as routing messages among robots, with special regard to messages to or from the coordinator entity.
Finally the uppermost layer consists of an application layer protocol that defines message formatting and exchanging. The algorithm ruling the asynchronous election of a leader among the robots in the team (in order to manage task assignments, among other things) will be implemented here.
The following sections will provide further details on each of the mentioned layers.

4. The Communication Layer

The lower layer includes the physical communication modules, which supervise the communications among robots and between robots and wireless sensor nodes. Moreover, this layer contains the modules necessary for the low-level functions of the localization and proximity networks.
A wireless LAN will act as a backbone and provide connectivity to the robots, while limiting modifications to the environment to the deployment of few access points, with no cabling, in order to adhere to the previously mentioned requirements; robots will use the standard IEEE 802.11a (WiFi) protocol to communicate among themselves, with the possibility of using broadcast or multicast addressing schemes, besides simple unicast. Remote access from the Internet to the functionalities provided by the system will also be possible through this backbone network.
A specialized wireless sensor network is thoroughly deployed to assist robots in their self-localization phase, according to the specifications of the Cricket project (Priyantha et al., 2000), this localization network is composed of Mica2 motes mounting an Atmega 128L 8-bit processor with 8 kBytes of RAM, 128 kBytes of FLASH ROM, and 4 kBytes of EEPROM and equipped with a CC1000 RF transceiver and an ultrasonic transmitter and receiver.
A few of those nodes are located at fixed positions in each sub-environment, while another one (named *listener*, in Cricket terminology) is carried by each moving robot. The same kind of nodes will also be carried by tourists, but, rather than acting as listeners, will only exploit their RF transceiver in order to provide proximity estimates.
In ideal conditions, the energy of the RF signal decreases with the square of the distance from the emitter, and this information could be used by a receiver to estimate its distance from the source of the signal as a function of the strength of the received signal. However, radio propagation is likely highly non-uniform in real environments, so received signal strength indicators (RSSIs) suffer from noisy measurements and distance predictions using signal strength are somewhat imprecise.
Nevertheless, proximity measures may still be used as an inexpensive means for the robots to approximately estimate how many tourists are “following” them and to plan the guided tour accordingly. Moreover, the same proximity sensors may be placed at specific spots (e.g. at each passage between contiguous sub-environments) in order to gather statistics about tourist flows by storing the IDs they sense over time for surveillance purposes.
Since robots are also equipped with standard WiFi cards in order to communicate with the above mentioned backbone network, they can communicate with Mica2 nodes on one side, and with the WiFi access points on the other, and are thus natural candidates for harvesting data from proximity sensor nodes (which do not form a connected network).

5. The Intermediate Layer

Data originating from the different kinds of networks described above will be processed at the middle layer of the RoboNet architecture, in order to convert application-layer messages into physical-layer ones, also adapting them to the requirements of the specific network that will carry them, to assist robots in the localization phase, and to extract proximity information about visitors.

Our framework achieves coordination among robots by a specifically designed application-layer protocol whose messages are carried by the WiFi backbone.

Application-layer messages will travel over UDP, managed by a simple loss recovery mechanism implemented as a thin sub-layer; using TCP would have led to unacceptable latency, and bare UDP would have suffered from the high loss rates typical of a wireless network such as the one considered here. After a message is sent, an ACK is expected and, if a pre-defined timeout expires before the ACK is received, the sender tries a retransmission and waits for twice the timeout; in case of further failure, a loss is simply notified to the upper layer.

Localization is realized by combining data coming from the analysis of internal robots' parameters and from the Cricket infrastructure. As described in (Balakrishnan et al., 2003), the most recent version of the Cricket framework implements a Kalman filter to assist a moving device in tracking its position. Cricket mobile nodes initially estimate distances from nodes at known positions by measuring the time difference of arrival between RF and ultrasound signals; the Kalman filter uses a predictor to estimate of the node's current position and corrects this estimate taking into account the difference between the predicted and the sensed distance. A covariance matrix reflects the filter’s confidence in the state vector.

Following the directions of (Smith et al., 2004), we provide each moving robot with an *active* localization device (the listener), so beacon nodes, whose location is known, estimate distances to the listener based on an active transmission from the listener itself.

Since a mobile device in the active mobile architecture sends simultaneous distance estimates to multiple receivers, its performance is arguably better than with the passive mobile system in which the listener obtains only one distance estimate at a time and may have moved between successive estimates.

As the listener is bolted to the robot whose movements are constrained in two dimensions, we modify the original EKF used to track the mobile device (Smith et al., 2004) considering a state vector composed by the two locations coordinates in the plane and the heading direction (θ). Moreover to take into account the information provided by the control data (i.e. the translational velocity v_t, and rotational velocity ω_t applied to control the robot), we adapted the prediction step of the EKF including a velocity motion model (Thrun et al., 2005). The state prediction at time t is:

$$\bar{\boldsymbol{\mu}}_t = \boldsymbol{\mu}_{t-1} + \begin{pmatrix} -\frac{\hat{v}_t}{\hat{\omega}_t}\sin(\theta) + \frac{\hat{v}_t}{\hat{\omega}_t}\sin\left(\theta + \hat{\omega}_t \Delta t\right) \\ \frac{\hat{v}_t}{\hat{\omega}_t}\cos(\theta) - \frac{\hat{v}_t}{\hat{\omega}_t}\cos\left(\theta + \hat{\omega}_t \Delta t\right) \\ \hat{\omega}_t \Delta t \end{pmatrix} \qquad (1.1)$$

where $\boldsymbol{\mu}_t$ is the predicted state at time t and $\boldsymbol{\mu}_{t-1}$ is the state vector at time $t-1$; v_t and ω_t are the translational velocity and the rotational velocity respectively, generated adding Gaussian noise to the motion control $\mathbf{u}_t = (v_t\ \omega_t)^T$.
The estimated pose provided by the EKF is integrated with the one generated using a previously developed particle filter algorithm (Thrun, et al., 2005).
In particular the posterior distribution of the robot pose is computed as a weighted sum:

$$p(x_t \mid z_t, u_t) = w_{wsn,t}\, p(x_t \mid z_{wsn,t}, u_t) + w_{rs,t}\, p(x_t \mid z_{rs,t}, u_t) \tag{1.2}$$

where $p(x_t|z_{wsn,t}, u_t)$ is the normal posterior distribution computed with the EKF taking into account measurements provided by the WSN ($z_{wsn,t}$) and $p(x_t|z_{rs,t}, u_t)$ is the posterior distribution computed by the particle filter given the robot sensors measurements ($z_{rs,t}$).
The weights $w_{wsn,t}$ and $w_{rs,t}$ are a measure of the uncertainty of the corresponding pose estimates. In particular $w_{wsn,t}$ is the likelihood $p(z_{rs}| x_t)$ of the measurement model (Thrun et al., 2005), while $w_{rs,t}$ is computed as the inverse of the trace of the covariance Σ_t of the posterior distribution estimated by the Kalman filter.
The original localization algorithm may thus be customized to our scenario allowing better performance.

6. The Application Layer

Robots belonging to the team need to agree on an initial representation of the world they operate in; moreover, they are not supposed to perform tasks independently from each other, and need some degree of coordination. The uppermost layer of the proposed architecture deals with providing the robots with proper representation of the environment, and implements a robust task assignment algorithm.
The present work builds upon a previous experience on the same topic (Chella, et al., 2007); in the referenced work, the proposed coordination mechanism relied on a central coordination unit that contained a complete knowledge of the environment and supervised the task assignment job through an auction mechanism. On the other hand, in the current version we abandoned this centralized approach as it introduced a potential bottleneck and a single point-of-failure, and instead we devised a fully distributed control where each robot is a potential coordinator, and this role will be assigned dynamically through a simple election mechanism.
We assume the backbone network is fully connected, which appears reasonable given the communication range of WiFi devices and the relatively short distances of an indoor environment. At the Intermediate Layer, the internetworking subsystem, transparently to the robots, builds a communication structure that allows addressing each of the robots singularly; additionally, multicast addressing is possible.
At the Application Layer, communications among robots occur via the backbone network that acts as a bus, and a simple algorithm is implemented for electing one of the robots as the temporary coordinator, following well-known techniques from related literature (Santoro, 2006). Roughly speaking, the algorithm guarantees that, at any given time, exactly one of the robots will act as the coordinator for the whole. When a leader must be elected, all robots will broadcast a “coordinator election request” message on the communication network; each message will contain the sender’s identification number, and the highest ID will eventually be used to have all robots agree on an elected coordinator. As will become clear in Section 6.1, the particular addressing scheme employed in our framework allows for

some flexibility with respect to each robot's ability of acquiring and releasing the coordinator role.
Such algorithm is clearly fully decentralized, and it can be proved that it is also asynchronous and fault-tolerant. Further details can be found in (Santoro, 2006).
In the following, the coordinator will be indicated by the term Robot Team Leader (RTL); it is important to point out that this role is not statically assigned to one of the robots, but will be taken up by several of them in the course of operations.
For the purpose of the operations of the RTL, the environment is modeled as a topological map representing the connectivity and accessibility of the different regions in the environment, similarly to what described in (Chella, et al., 2007). The environment is thus split into a collection of sub-environments connected by passages, and the relations of connectivity between the different sub-environments are captured by a connectivity graph representation: a node corresponds to a sub-environment; each sub-environment is univocally identified; an arc between two nodes exists if the corresponding sub-environments are connected.
The condition of accessibility for each robot is modeled by partitioning the connectivity graph and considering the subgraph representing regions reachable by the robot.
A visit is described in the model by the sequence of sub-environments to be showed to visitors. For the reasons stated above, a visit generally needs to be split into a sequence of sub-visits each one guided by a robot. The RTL governs the assignment of sub-visits to robots by an auction mechanism.
When a guiding service is needed the RTL is notified by either the ticket counter or a robot, with the former case happening only once at the beginning of a visit, while the latter may arise several times during the visit. In fact, every time a robot is not able to reach the next sub-environment, another robot must be found to lead the visitors group throughout the remaining part of the visit.
When a robot decides to cease guiding the group, for instance because it cannot physically reach the next sub-environment, it communicates the remaining part of the visit to the RTL. This mechanism allows for the inclusion of reasons other than the sub-environment connectivity into the management of guided tours; for instance drawn batteries as well as other malfunctions could be easily circumvented, given other robots were allocated to the same sub-environment. In practice, a request from a robot guide that ceased a visit is identical to those issued by the ticket counter for a new visit and both are encoded with the same message kind in the protocol.
When the RTL receives such a message it starts the auction by multicasting the request for a task to the robots able to reach the first sub-environment of the visit. The robots then reply to the RTL with their bid expressed as an estimate of the time needed to start the requested visit. This estimates accounts for the time to get to the starting point as well as the remaining time of the current visit, if any.
The current RTL collects the bids and sends the robot with the best bid a task assignment message. Encoded in the message, beside the sequence of sub-environments to be shown, are the IDs of the visitors. The receiver can then reply to accept the task thus ending the auction, otherwise another bidder must be chosen by the RTL.
A visit may end naturally when the last sub-environment has been visited or when the robot guide senses no visitors in its proximity.
The leadership alternation approach we adopted allows to carry out an intrinsically fault-tolerant system with respect to the failures of a single central unit, while in the meantime the benefits of a centralized coordination approach are maintained.

Each robot is able to perform as a leader as soon as it is selected. To this aim, both the topological map of the whole environment and its partitions (representing the accessibility of sub-environments for each robot) are initially provided to all the robots and represent the *innate* knowledge of the system.
The proposed framework has been designed to allow for an easy extension process of the innate system knowledge with new information acquired during run-time. In particular, we analyzed the possibility of modifying the auction mechanism by sending the request for a task assignment to a subset of the robots that are able to perform it. Such subset can be selected by performing a statistic of the past bids made from the robots in a similar case.
The evolution of the system knowledge through learning, combined with the mechanism of leader alternation at the same time point out the issue of knowledge sharing. To this aim the proposed framework allows to verify different approaches:

- knowledge sharing and updating every time the leader changes;
- no knowledge sharing.

In the first case, whenever a new leader has to be elected, the current leader broadcasts the new information to all the robots, so achieving a common updated and consistent knowledge representation.
The second approach will lead to different leaders, depending on the information they received during their leadership. This approach may for instance allow for easy checking of the performances of different learning algorithms as a function of the dynamics of the acquisition of new knowledge about the environment in the course of time.
Moreover this approach allows to bound the drawbacks due to the over fitting produced by some learning algorithms. For example one leader can decide to leave out a robot from the auction related to a certain task. The diffusion of such information could produce the erroneous exclusion of this robot from future auction even if the temporary failure that affected the robot has been solved.
Although such issues can be addressed by performing a periodic query of the robots state, the local nature of certain information allows to naturally fit the system to variable conditions. However a global update of the system knowledge can be periodically performed to provide each robot with the possibility of evolving and growing its own knowledge.

← 16 bit →

<table>
<tr><td colspan="2">Msg Type</td><td colspan="3">Message ID</td></tr>
<tr><td>lb</td><td colspan="2">Source ID</td><td>mc</td><td>Destination ID</td></tr>
<tr><td colspan="5">Payload Length</td></tr>
<tr><td colspan="5">…
Data
…</td></tr>
</table>

Fig. 2: The message format.

6.1 The Message Exchange Protocol

We devised an application-layer protocol, adapted with minimal changes from (Chella, et al., 2007) in order to allow the RTL to send and receive requests from robots and the robots to communicate with each other. Messages will travel on the WiFi network and have the format specified in Fig. 2, which shows a fixed 6-bytes header followed by a variable length payload.

Message Type	Description
LDR_SEL	used during the leader election phase
RBT_PING	ping message to check for liveliness
RBT_PONG	response to ping
SVC_SIG	used to signal the RTL about a new visit to be assigned
SVC_REQ	used by RTL for starting the auction
SVC_REPLY	used by a robot to participate in the auction, and signal its service response time
TASK_ASS	used by RTL to assign a task to a robot
TASK_ACC	used by a robot to confirm acceptance of a task
SYNC_REQ	used by a robot to exchange its state with another robot
SYNC_REPLY	reply to SYNC_REQ

Table 1: Message types.

The semantics of carried data depend on the specific message type that is specified in the 4-bit "Msg Type" field; this is more than sufficient to identify one of the different types we are currently using for messages, as will be detailed in the following. Messages traveling across the network will be uniquely identified by the combination of a 12-bit "Message ID" and of a "Source ID".

Each robot is statically assigned an ID, which will be also used to identify the source of a message, indicated in the "Source ID" field as a 7-bit integer (the first bit is reserved for the leader election mechanism). The next 8 bits carry the message destination information; this field is 7-bit long, with 1 preceding bit reserved for multicast groups management; when set to 1, it indicates a multicast address (in order to identify specific groups of robots), while referring to individual robots otherwise. This "Destination ID" field contains the unique identifier of any addressable entity in the RoboNet framework, namely every individual robot or any predefined group of robots. The address 0000000 is reserved and will be used in certain circumstance by the RTL to identify itself instead of its natural address.

We allow for a maximum of 2^7-1 different *RobotID*'s, thus limiting to 127 the amount of robots in the environment, a number that appears sufficient for any practical purpose.

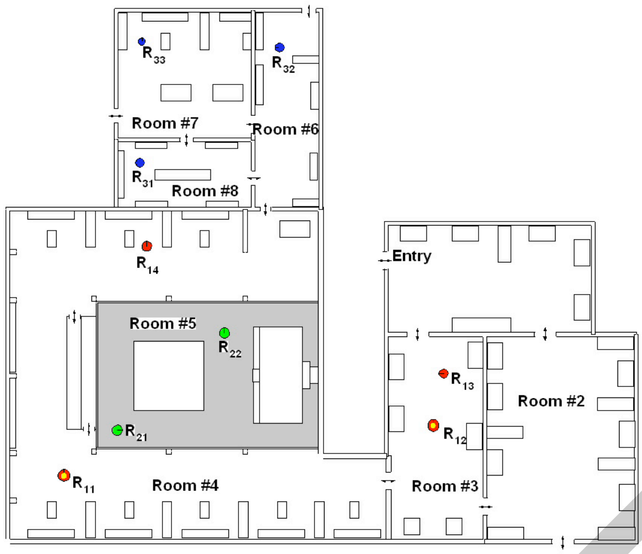
Unique IDs are also assigned to rooms and passages between rooms; moreover, as each visitor will be provided with a mote acting as a tracking device, they may also be uniquely identified through that device's ID; such ID's will be stored in a 16-bit long field to be carried in the payload of the message; the maximum amount of elements for those *IDList*'s is only limited by the message payload length, specified in the relative field. In the following, such ID's will be indicated respectively as *RoomID*, *PassageID*, and *VisitorID*.

As a group forms at the entrance, a list of *VisitorID*'s is created and is transmitted to the current RTL in a SVC_SIG; RTL will then broadcast a SVC_REQ message to all robots through the WiFi backbone. The message will include the room where service is required (i.e. the starting room for the tour). Each robot will reply with a SVC_REPLY message containing an estimate for its service time, as explained above. RTL will then assign the task to one of the robots whose reply has been received and that provided the best bid, and the corresponding TASK_ASS message contains the list of *VisitorID* and the list of *RoomID* (i.e. the description of the "visit"). The selected robot will finally signal its acceptance by sending a TASK_ASS message to RTL.

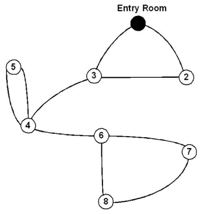
Finally, the `LDR_SEL` message is used during the leader election phase. This message will be broadcast by each robot periodically, either spontaneously or upon reception of an analogous message from one of its fellows. The robot with the highest ID will eventually be selected as the leader, and this information will be automatically shared by the whole team.
The current leader, after a fixed period of time, broadcasts a `LDR_SEL` message to all the other robots to signal its will to resign from its role in favor of a potential new leader, and will thus trigger a new election. We assume that each robot knows the IDs of the other participants to the team, so that after receiving all the answers it can infer whether it won the race to leadership or not. A timeout mechanism provides robustness: if a robot believes it is next elected leader, but has not received answers from some other robot within the timeout, it will simply check their status through a `RBT_PING` message, and will consider them "dead" for the current round. After a grace period, necessary to the old leader to end any previous assignments, the new elected leader will be effectively operating.
This mechanism is RTL-driven, so it could be prone to error as a consequence of a fault in the current RTL; for this reason, the protocol assumes that each robot periodically pings the leader to check if it is still alive and functioning. Upon detection of a fault, the robot that noticed the anomaly independently starts the leader election mechanism.
As previously mentioned, the current RTL is identified by the `000 0000` reserved address, which would "force" it to resign from its role, as a consequence of the "highest ID wins" rule. The "leader boost" bit preceding the Source ID field may be used to artificially modify the natural address in order to allow for some flexibility in the election mechanism, according to a policy that can also be defined at the application layer. An example of this will be given with reference to the application scenario described later on; for instance, the current RTL may force its address to **1**`000 0000` in order to signal its availability to maintain the leadership. Analogous behavior is implemented for non-leader robots.
The complete set of messages is shown in Table 1, together with a brief description.

7. An Application Scenario

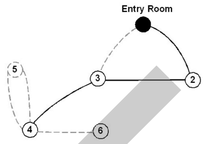
In the course of the years, the Robotics Lab of University of Palermo developed a robotic architecture that takes into account several suggestions from cognitive science.
The architecture has been successfully tested in the CiceRobot project on tasks related to guided tours in the Archaeological Museum of Agrigento (Chella & Macaluso, 2006; Chella et al., 2007). The robot was able to modify the predefined paths through the museum by rearranging the sequence of findings to be visited depending on user queries.
For the purpose of the current research, it is important to point out that the robot planning system is based on a 3D Robot/Environment Simulator. The planning by simulation paradigm allows to easily and carefully perform those forms of planning that are more directly related to perceptual information. In fact, the preconditions of an action can be simply verified by geometric inspections in simulation, avoiding the verification by means of logical inferences on symbolic assertions; also the effects of an action are not described by adding or deleting symbolic assertions, but they can be easily described by the situation resulting from the expectations of the execution of the action itself in the simulator.
The proposed architecture has been deployed in the Archaeological Museum of Agrigento. As previously said, each robot is able to guide visitors both in a prearranged tour and in an interactive tour. Let us consider the multi-robot coordination in our experimental setup.



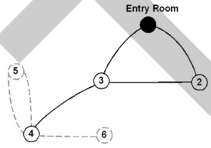
(a) Map of the environment.



(b) Graph representation of the whole environment.



(c) Graph representation of the environment for Robot R_{13}.



(d) Graph representation of the environment for Robot R_{12}.

Fig. 3: The site described in the application scenario.

Supposing that a group of visitors is waiting at the entrance; the RTL multi-casts a `SVC_REQ` message to those robots that are able to reach the entry (red and yellow/red robots in Fig. 3). Each robot replies with its estimated performance relative to the current task, i.e. the time it takes to reach the entry. Through the 3D Robot/Environment simulator each robot can imagine itself going through the environment to reach the current target: the simulated interaction between the robot and the environment allows to easily compute an accurate estimate of the robot performance. In the example reported in Fig. 3 both robots R_{11} and R_{14} are busy guiding visitors, while robots R_{12} and R_{13} are idle. Even if R_{13} is the nearest robot to the entry, the RTL allocates the task to the robot R_{12} as its performance estimate is better. Actually R_{13} is not able to directly go from *Room #3* to *Entry Room* due to a slope that is too steep for it, but not so for R_{12}. During the visit, as R_{12} is about to complete the sub-visit it has been assigned, it sends a `SVC_SIG` message to the RTL to request another task allocation for the group.

We also performed several tests on a simulated environment to evaluate the performances of the leadership alternation approach.

It should be pointed out that the leader is functionally identical to the other robots, except for the coordination burden it must deal with. Therefore, the leader too participates to every auction for tasks assignment; anyway to reduce its computational load, the bid proposed by the leader is incremented by an additional cost. This way we reduced the number of tasks the leader will self-assign. Moreover, in agreement with the need of reducing the leader computational load, the participation to the election phase can be restricted to the robots that are currently idle; they may reinforce their proposal for leadership by setting the "leader boost" bit in their address while sending the `LDR_SEL` message. No conflicts may

arise among robots competing for the leadership, as ties will be automatically broken by the use of unique addresses.
In some cases, the combination of the two mentioned mechanisms resulted in the unwanted persistence of the same leader. In order to overcome this drawback, the additional cost related to the leader tender is implemented as a linear decreasing function of time.
Fault tolerance is achieved in this setting through the periodic check on the leader status performed by all robots, at the cost of a minimal traffic overhead introduced by the control messages.
At the moment of writing, the experiments were performed assuming no knowledge sharing was carried out by robots. More elaborated scenarios could be devised so that the information acquired so far by the current leader is passed onto its successor and merged in order to achieve a common updated and consistent knowledge representation.

7. Conclusion

This paper presented a framework for the coordination of a group of robots moving in a structured indoor environment in order to manage automatically guided museum tours.
The design of a hybrid wireless networking architecture, composed by WiFi devices inter-operating with wireless sensor nodes has been discussed, and it has been shown how it can operate as a whole in order both to provide a communication backbone for the robots, and to extract useful information from the environment.
The robustness of the communication protocol implemented in the proposed framework has been enforced through a fault-tolerant leader election mechanism, which allows for an easy extension process of the innate system knowledge with new information acquired during run-time.
Experiments have been carried on in the context of the RoboNet project conducted at the Archaeological Museum of Agrigento, Italy, and the proposed coordination mechanisms have been tested through simulations.

8. References

Akyildiz, I.; Su, W.; Sankarasubramaniam, Y. & Cayirci, E. (2002). A survey on sensor networks. *IEEE Communication Magazine*, vol. 40, n. 8, August 2002, pp. 102–114.

Alami, R.; Fleury, S.; Herrb, M.; Ingrand, F. & Robert, F. (1998). Multi-robot cooperation in the MARTHA project. *Robotics & Automation Magazine, IEEE*, vol. 5, no. 1, pp. 36–47.

Balch, T.; Arkin, R. C. Communication in reactive multiagent robotic systems. Autonomous Robots 1:1-25. (1994)

Balakrishnan, H.; Baliga, R.; Curtis, D.; Goraczko, M.; Miu, A.; Priyantha, N. B.; Smith, A.; Steele, K.; Teller, S. & Wang., K. (2003). Lessons from developing and deploying the cricket indoor location system. *Technical report, MIT Computer Science and AI Lab*.

Bergbreiter, S. & Pister, K. S. J. (2003) Cotsbots: An off-the-shelf platform for distributed robotics. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Las Vegas, NE, USA.

Chella, A.; Liotta, M. & Macaluso, I. (2007). Cicerobot - A cognitive robot for interactive museum tours. *Industrial Robot: An International Journal*, vol. 34, pp. 503-511 ISSN: 0143-991X.

Chella, A.; Lo Re, G.; Macaluso, I.; Ortolani, M. & Peri, D. (2007) Multi-robot Interacting Through Wireless Sensor Networks. *Lecture Notes in Computer Science*, vol. 4733, pp. 789-796, 2007. ISSN: 0302-9743.

Chella, A. & Macaluso, I. (2006) Sensations and Perceptions in ”Cicerobot” a Museum Guide Robot. *Proc. of BICS 2006 Symposium on Brain Inspired Cognitive systems*, Greece.

Dudek, G.; Jenkin, M.; Milios, E. & Wilkes, D. (1996) A taxonomy for multi-agent robotics. Autonomous Robots, vol. , no. 4, pp 375–397.

Mataric, M. J. (1997) Behaviour-based control: examples from navigation, learning, and group behaviour. Journal of Experimental & Theoretical Artificial Intelligence, vol. 9, no. 2, pp. 323–336.

Mataric, Maja J. (1998) Using Communication to Reduce Locality in Distributed Multi-Agent Learning, Journal of Experimental and Theoretical Artificial Intelligence, special issue on Learning in DAI Systems, Gerhard Weiss, ed., vol 10, no. 3, pp. 357–369.

McMickell, M. B.; Goodwine, B. & Montestruque, L. A. (2003) Micabot: A robotic platform for large-scale distributed robotics. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Taipei, Taiwan.

Moore, K. L.; Chen, Y. Q. & Song, Z. (2004) Diffusion based path planning in mobile actuator-sensor networks (mas-net) - some preliminary results. *Proc. of SPIE Conf. on Intelligent Computing: Theory and Applications II, part of SPIEs Defense and Security*, Orlando, FL, USA.

Parker, L. E. (2003) Current Research in Multirobot Systems, Artificial Life and Robotics, vol. 7, no. 2-3.

Priyantha, N. B.; Chakraborty, A. & Balakrishnan, H. (2000) The cricket location-support system. *Proc. of the Sixth Annual ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Boston, MA, USA.

Santoro, N. (2006) Design and Analysis of Distributed Algorithms (Wiley Series on Parallel and Distributed Computing. ISBN: 978-0-471-71997-7.

Smith, A.; Balakrishnan, H.; Goraczko, M. & Priyantha, N. B. (2004) Tracking moving devices with the cricket location system. *Proc. of the 2nd ACM MobiSys*, pages 190–202, Boston, MA, USA.

Thrun, S.; Burgard, W. & Fox, D. (2005) Probabilistic robotics. MIT Press.

Winfield, A. (2000) Distributed sensing and data collection via broken ad hoc wireless connected networks of mobile robots. In: Parker L, Bekey G, Barhen J (eds) Distributed autonomous robotic systems, vol 4. Springer-Verlag, Tokyo, pp 273-282