Augmenting the Physical Environment Through Embedded Wireless Technologies

Marco Mamei[1] and Franco Zambonelli[1]

Dipartimento di Scienze e Metodi dell'Ingegneria,
University of Modena and Reggio Emilia
Via Allegri 13, 42100 Reggio Emilia, Italy
{mamei.marco, franco.zambonelli}@unimo.it

Abstract. Emerging pervasive computing technologies such as sensor networks and RFID tags can be embedded in our everyday environment to digitally store and elaborate a variety of information about the surrounding. By having application agents access in a dynamic and wireless way such distributed information, it is possible to enforce a notable degree of context-awareness in applications, increase the capabilities of interacting with the physical world, and eventually give a concrete meaning to the abstract concept of agent situatedness. This paper discusses how both sensor networks and RFID tags can be used to that purpose, outlining the respective advantages and drawbacks of these technologies. Then, to ground the discussion, it presents a multiagent application for physical object tracking, facilitating the finding of "forgot-somewhere" objects in an environment.

1 Introduction

The never ending technological progresses in miniaturization of electronic devices and in wireless communication technologies are making possible to enrich our everyday environments (and any objects in them) with sensing, computation, and communication capabilities [?]. Overall, this may end up in an increased capability of interacting with the physical world, by acquiring in a digital form and in a wireless way a number of information beyond the normal sensing capabilities of humans and robots, as well as in the possibility of exploiting the devices embedded in the environment as a pervasive platform for distributed computing and communication.

With reference to the multiagent systems paradigm and to agent-oriented software engineering [?], the advent of such pervasive computing technologies notably impacts on the concept of *situatedness*. Agents has always been assumed – by very definition [?] – as entities whose activities are related to some sensing and effective of the properties of some environment in which they situate for execution. However, despite this, the concept of environment has always been an under-considered topic, and a few proposals for agent languages and architectures explicitly deal with this concept in a constructive way [?]. Persavive computing technologies, by making available to application agents expressive

digital information about the environment, can leverage the concept of situatedness from a mere conceptual definition to a practical useful feature.

Starting from the above considerations, this paper to discuss which technologies can be actually be used to this purpose. In particular, this paper shortly present both sensor networks technologies [?] and RFID technologies [?], and discuss how these technologies can be exploited to augment physical environments with the possibility of easily accessing digital information about them, as well as with the possibility of enforcing forms of stigmergic (i.e., environment-mediated) interactions across the physical environment itself. A comparative analysis of these two technologies outlines their respective advantages and limitations, and their potentials in pervasive multiagent systems applications.

To ground the discussion, we presents our own experience in the implementation of a multiagent application for stigmergic physical object tracking, allowing agents (whether in the form of autonomous robots or computer-assisted humans) to find "forgot-somewhere" objects in an environment. The application relies on pheromone-based interaction, and exploit RFID tags as a physically distributed memory infrastructure in which agents can deploy pheromones and that agents can access for reading pheromone path spread in the environment.

The following of this paper is organized as follows. Section 2 introduces in general the concept of situatedness and the novel scenarios opened by adoption of pervasive computing technologies for situated multiagent applications. Section 3 presents and discussed sensor network technologies, while section 4 presents and discusses RFID technologies. Section 5 presents our experience in RFID-based object tracking. Section 6 concludes and outlines open directions.

2 Situatedness and Physical Environments

While the concept of situatedness plays a fundamental role in multiagent systems engineering, the practical application of the concept cannot abstract from what actual infrastructures are available to model the environment and to interact with it. In this section, after having discussed the various facets of situatedness, we analyse how pervasive computing technologies can be used to somehow "augment" a physical environment to facilitate agents in interacting with it.

2.1 Computational vs. Physical Environments

Software systems are rarely developed to be deployed as stand-alone, isolated systems. Rather, in most practical cases, software systems (and multiagent systems specifically) are designed and deployed for being deployed in some sort of existing computational or physical environments, and have to necessarily interact with such environments to properly accomplish their tasks [?].

As far as computational environments are concerned, modern distributed applications are always built to interact with an existing world of data, services, and computational resources, and have to get advantage of them. For instance, in multiagent systems for Web-based applications agents are deployed in the Web

and are made in charge of mining Web data and exploiting available services in order to achieve specific goals (e.g., organizing a trip for the user by discovering appropriate tourist information and by booking flights and hotels as needed) [?]. In the Grid, agents have to interact and negoatiate for access to computational and memory resources [?]. In P2P systems, a networks of autonomous components (that can be assimilated to agents) have to connect and interact with each other in order to provide access to large set of shared files [?,?].

As far as physical environments are concerned, mobile and pervasive computing technologies are making ICT systems more and more strictly inter-twined with the physical world. Firstly, mobile computing technologies, enabling us to stay connected 24/7 from wherever, require context-awareness and context-dependency, to have our computer-supported activities properly adapted to the physical context and situation from which we are performing them [?]. Secondly, more and more autonomous software systems (or, which is the same, systems of autonomous robots [?]) are deployed to monitor and/or control processes occurring in the physical world, e.g., system for control of manufacturing processes [?] or of human activities [?].

2.2 Environment-mediated Interactions

Whatever the specific case one considers, it is rather clear that the design and development of a multiagent system cannot abstract from the characteristics of the computation environment in which it will be deployed. Thus, the necessity of exploiting an environment abstraction necessarily comes into play simply because such environment exists in a computational form, which does not relief designers and developers from a careful modeling effort, as introduced in the previous subsection.

Also in the case of an existing computational environment, it is necessary to properly study the characteristics of such environment, of its data and of its services, and model it to capture what's necessary from the viewpoint of the developing system. Thus, such modeling activity may imply identifying what are the environmental entities that need to be taken into account, identifying how the agents of the developing multiagent system can (if at all) access them, and identifying the dynamics of such environment, to make the agents of the developing systems ready to face such dynamics.

In addition, a modeling effort may be requested to properly identify what in the existing environment should be characterized as environmental resources and what, instead, as agents with which to interact. In some cases, such identification may be trivial: considering the case of an on-line auction market [?] , environmental resources (e.g., goods to be sold and their description) are clearly distinguished from agents (the auctioneer and competing bidders). In other cases, the distinction may be less sharp (consider, e.g., the case of an active push-based database), and some identification and modeling effort, possibly leading to the agentification of some environmental entities, may be needed.

In any case, the key message is that the environment in which a multiagent system is to be deployed should not be considered as given. Rather, its char-

acteristics must be explicitly identified, modeled, possibly shaped as needed to reduce complexity and facilitate deployment, as well as to bring in the already outlined advantages in terms of modularity and separation of concerns.

The software engineering impact of the above technologies is that software systems (i.e., the agents in a multiagent system) have to carry on their activities by continuously processing – and being continuously affected by – what is happening in the physical world. In other words, it is like software systems executed by being immersed in a physical environment, and by having to interact with its (virtualized) resources. On this base, we could copy here all the discussion we have already made with regard to computational environments: the need of modeling such physical environmental resources, of properly identifying their dynamics, and possibly the need of agentifying some of the environmental entities (e.g., the active sensors of a sensor network). And, also in the case of physical environments, we can re-state that the introduction of the environment abstraction is not only a software engineering need, but also a driver for reducing complexity and enforcing modularity and separation of concerns. Fields - pheromones.

Active - passive infrastructure

3 Ad-Hoc and Sensor Network

As proved in the context of the Smart Dust project at Berkeley [?,?], it is already possible to produce fully-fledged computer-based systems of a few cm^3, and even much smaller ones will be produced in the next few years. Such computers, which can be enriched with communication capabilities (radio or optical), local sensing (e.g., optical, thermal, or inertial) and local effecting (e.g., optical and mechanical) capabilities, are the basic ingredients of the sensor network scenario (see figure 1).

Such a scenario implies spreading a large number of these sensing devices across an environment, letting them create an ad-hoc wireless network by communicating with each other and perform some kind of distributed application. Traditional applications can vary from monitoring of physical parameters (e.g., monitoring weather) and distributed surveillance (e.g., tracking vehicles crossing a specific area).

3.1 Deploying Digital Information

In general terms, sensor networks are also an ideal platform to augment the physical environment with digital information.

- Sensors can store data to represent some kind of contextual information. Moreover, they can deliver such data to agents (e.g., users with PDA) passing nearby.
- Sensors can perform computations to support and facilitate the agents' fruition to that data. For example, sensors can propagate and diffuse data

Fig. 1. Wireless sensor devices

across the network. They can automatically delete old and possibly corrupted information. They can combine and transform data to let it become more expressive and easy to use.

The distributed nature of the sensor network allows to augment the digital world with distributed overlay data structures providing agents with effective, flexible and easy-to-be-accessed information.

Overlay data structures are distributed data structures encoding specific aspects of the agents' operational environment. These overlays cab be propagated across a sensor network in order to be easily accessible by the agents and to provide easy-to-use context information (i.e., the overlays are specifically conceived to support their access and fruition). The strength of these overlay data structures is that they can be accessed piecewise as the application agents visit different places of the distributed environment. This lets the agents to access the right information at the right location. Overlay data structures can be also dynamically spread by an agent in order to represent and "communicate" its own activities. From this point of view, overlay data structures enable "stigmergy" [**?**] in that agents' interactions can be mediated by these kind of overlay "markers" distributed across the environment.

cosa ci facciamo –¿ ambiente distribuito attivo dove propagare info che posoono essere acedute tiop spazi di tuple, fields, feromoni, self-localization

scaricare compiti dall'agente all'ambiente.

ambiente fornisce info pre-elaborate all'agente

red carpet

l'ambiente offre informazionifacilmente capibili e utilizzabili al agente

vantaggi e svantaggi (costo, batterie, costo-deployment, si richiede un infrastruttura)

In the next section we will detail another technology that tries to solve some of this problems.

4 RFID Technology

Advances in miniaturization and manufacturing have yielded postage-stamp sized radio transceivers called Radio Frequency Identification (RFID) tags that can be attached unobtrusively to objects as small as a toothbrush. The tags are wireless and battery free. Each tag is marked with an unique identifier and provided with a tiny memory, up to some KB for advanced models, allowing to store data (our test-bed implementation comprise tags with a storage capacity of 512bit). Tags can be purchased off the shelf, cost roughly 0.20 Euro each and can withstand day-to-day use for years (being battery-free, they do not have power-exhaustion problems). Suitable devices, called RFID readers, can access RFID tags by radio, either for read and write operations. The tags respond or store data accordingly using power scavenged from the signal coming from the RFID reader. RFID readers divide into short- and long-range depending on the distance within which they can access RFID tags. Such distance may vary from few centimeters up to several meters. Given this technology, our scenario requires that a number of places in the environment (e.g. doors, corridors, etc.) or unlikely-to-be-moved objects (e.g. beds, washing machines, etc.) are tagged with RFID tags. Tagging a place or an object involves sticking an RFID tag on it, and making a database entry mapping the tag ID to a name. In the following of this paper, we will refer to these tags as location-tags. RFID readers accessing one of these tags can lookup the tag ID into the database and infer to be close to the corresponding place. It is worth noting that, other than the pheromone-deployment mechanisms described in the following, such basic technology could implement a primitive localization mechanism [**?**]. As a final note, it is worth emphasizing that current trends indicate that within a few years, many household objects and furniture may be RFID-tagged before purchase, thus eliminating the overhead of tagging [**?**]. Moreover, some handheld devices start to be provided with RFID read and write capabilities (the Nokia 5140 phone can be already equipped with a RFID reader [**?**]).

4.1 Deploying Digital Information

As anticipated in the introduction, pheromones are created by means of data-structures stored in RFID tags. The basic scenario consists of human users and robots carrying (embedding) handheld computing devices, provided with a RFID reader, and running an agent-based application. The agent, unobtrusively from the user, continuously detects in range location-tags as the user roams across the environment. Moreover, the agent controls the RFID reader to write pheromone data structures (consisting at least in a pheromone ID) in all the tags encountered. This process creates a digital pheromone trail distributed across the location-tags. More formally, let us call L(t) the set of location-tags being sensed at time t. It is easy to see that the agent can infer that the user is moving if $L(t) \neq L(t\text{-}1)$. If instructed to spread pheromone O, the agent will write O in all the L(t)-L(t-1) tags as it moves across the environment. For the majority of applications a pheromone trail, consisting of only an ID, is not very useful. Like

in ant foraging, most applications involve agents to follow each other pheromone trails to reach the location where the agents that originally laid down the trail were directed (or, on the contrary, to reach the location where they came from). Unfortunately, an agent crossing an-only-ID-trail would not be able to choose in which direction the agent that laid down that trail was directed. From the agent point of view, this situation is like crossing a road without knowing whether to turn left or right. To overcome this problem, the agent stores in the location-tags also an ever increasing hop-counter associated with O - we will call this counter C(O). In particular, if an agent decides to spread pheromone O at time t, the agent reads also the counter C(O) in the L(t) set (if C(O) is not present, the agent sets C(O) to a fixed value zero). Upon a movement, the agent will store O and C(O)+1 in the tags belonging to L(t+1) that do not have O or have a lower C(O). In addition, the basic pheromone idea requires a pheromone evaporation mechanism to discard old - possibly corrupted - trails. To this end we store in the tag also a value T(O) representing the time where the pheromone O has been stored . To better understand how pheromone data structures are stored in RFID tags, it is fundamental to describe how the tag memory has been organized (see figure 1). RFID tags, other than an unchangeable unique identifier, are typically provided with an array of memory cells, each consisting of few bits. This is where pheromone data structures will be held. Our proposal is to organize such memory by allocating 3 slots for each pheromone. The first slot will hold the pheromone identifier. The second slot will hold the associated counter. The third will hold the above mentioned timestamp. The very first slot in the tag will be used to hold an index pointing to the first slot available for pheromone storage. Since RFID tags are completely passive devices, upon a pheromone insertion, the RFID reader must: (i) read the tag index from tag[0]; (ii) store in tag[index] the pheromone id; (iii) store in tag[index+1] the pheromone counter; (iv) store in the tag[index+2] the timestamps; (v) store the new index+3 in tag[0].

To read pheromones, an agent trivially accesses neighbor RFID location-tags reading their memories. Since RFID read operations are quite unreliable, the agent actually performs a reading cycle merging the results obtained at each iteration. Given the result, the agent will decide how to act on the basis of the perceived pheromone configuration. To realize pheromone evaporation, after reading a tag, an agent checks, for each pheromone, whether the associated timestamp is, accordingly to the agent local time, older than a certain threshold T. If it is so, the agent deletes that pheromone from the tag. This kind of pheromone evaporation leads to two key advantages:

1. Since the data space in RFID tags is severely limited, it would be most useful to store only those pheromone trails that are important for the application at a given time; old, unused pheromones can be removed.
2. If an agent does not carry its personal digital assistant or if it has been switched off, it is possible that some actions will be undertaken without leaving the corresponding pheromone trails. This cause old-pheromone trails to be possibly out-of-date, and eventually corrupted. In this context, it is of

course fundamental to design a mechanism to reinforce relevant pheromones not to let them evaporate.

With this regard, an agent spreading pheromone O, will overwrite O-pheromones having an older T(O). From these considerations, it should be clear that the threshold T has to be tuned for each application, to represent the time-frame after which the pheromone is considered useless or possibly corrupted.

Fig. 2. Memory organization of RFID tags.

The key idea of our approach is to exploit the fact that RFID tags (in stark contrast with other tagging technologies like barcodes) can be written on-the-fly by suitable wireless devices, called RFID readers (which are also writers despite the name). On this basis, RFID readers, carried by a human or embedded in a robotic agent, could deploy pheromone trails across the environment, by storing the pheromone values in the RFID tags located there, as the user roams across the environment. The main point in favor of our approach is its extremely low price since it uses technologies (RFID) that are likely to be soon embedded in the scenario independently of this application. Relying on such an implementation, a wide range of application scenarios based on pheromone interaction can be realized ranging from multi-robot coordination [?], to monitoring of human activities [?]. In the followings, for the sake of illustrating our approach, we describe an application consisting in an agent-based system to easily find everyday objects (glasses, keys, etc.) forgot somewhere in our homes. In particular, the application allows everyday objects to leave virtual pheromone trails across our homes to be easily tracked afterwards.

5 Pheromone-based Object Tracking

In this section, we present a concrete application to test our approach. It consists in an agent-based application to easily find everyday objects (glasses, keys, etc.)

forgot somewhere in our homes. The application allows everyday objects to leave virtual pheromone trails across our homes to be easily tracked afterwards.

5.1 Overview

Overall, the object tracking application work as follows (details in the following subsection):

- The objects to be tracked need to be tagged. For sake of clarity, we will refer to these tags as object-tags to distinguish them from the location-tags identifying places in the environment.
- Agents (either robotic or humans) are provided with a handheld computing device, connected to a RFID reader, and running an agent-based application.
- The agent-based application can detect, via the RFID reader, object-tags carried on by the user. Exploiting the mechanism described in the previous section, it can spread a pheromone identifying such objects into the available memory of near location-tags (see section 3.2 for details).
- This allows the object to leave a pheromone trail across the location-tags in the environment.
- When looking for an object, a user can instruct the agent to read in-range location-tags searching the object's pheromone in their memory. If such pheromone is found, the user can follow it to reach the object current location (see section 3.3 for details).
- Once the object has been reached, if it moves with the user (i.e. the user grabbed it), the agent automatically starts spreading again the object associated pheromone, to keep consistency with the new object location.

This application naturally suits a multi-user scenario where an user (or a robot), looking for an object moved by another user, can suddenly cross the pheromone trail the object left while moved by the other user.

5.2 Spreading Object Pheromones

To spread pheromones, the agent needs first to understand which objects are currently being carried (i.e. moved around) by the user. To perform this task unobtrusively, it accesses the RFID reader to detect in-range RFID tags once a second. Let us call O(t) the set of object-tags being sensed at time t, L(t) the set of location-tags being sensed at time t. If the agent senses an object-tag O such that $O \in O(t)$, $O \notin O(t\text{-}1)$, but $L(t) \neq L(t\text{-}1)$, then the agent can infer that the user picked-up the object O and the object is moving around. In this situation, the agent has to spread O pheromone in the new location. To this end, the agent writes O in the available memory space of all the L(t) location-tags that do not already contain O. This operation is performed, for every object O, upon every subsequent movement. Similarly, if the agent senses that an object-tag $O \in O(t\text{-}1)$, but $O \notin O(t)$, then the agent infers that the user left object O. When this situation is detected the agent stops spreading O pheromone. These operations create pheromone trails of the object being moved around.

5.3 Tracking Objects

Once requested to track an object O the agent will start reading, once per second, nearby location-tags looking for an O-pheromone within the sensed location-tags L(t). If such a pheromone is found, this implies that the user crossed a suitable pheromone trail. There are two alternatives: either in L(t) there are two location-tags having O-pheromones with different C(O), or L(t) contains only one location-tag. In the former (lucky) case, the agent notifies the user about the fact it has crossed a pheromone trail and it suggest to move towards those location-tag having the higher C(O) . In the following, we will refer to this as grad-search, since it is like following a gradient uphill. In the latter (unlucky) case, the agent notifies the user about the fact it has crossed a pheromone trail, but nothing else. In such situation, the user has to move in the neighborhood, trying to find higher C(O) indicating the right direction to be followed (this is like dowsing – i.e. finding underground water with a forked stick – but it works!). In the following, we will refer to this as local-search. Following the agent advices, the user gets closer and closer to the object by following its pheromone trail, until reaching it.

6 EXPERIMENTS

To assess the validity of our approach and the effectiveness of the object tracking application, we developed a number of experiments, both adopting the real implementation and an ad-hoc simulation (to test on the large scale). Basically, our approach consists in developing a simulation matching the real implementation data, and then to use the simulation to extrapolate the results in large-scale scenarios.

6.1 Real Implementation Set-Up

The real implementation consisted in tagging places and objects within our department. Overall, we tagged 100 locations within the building (doors, hallways, corridors, desks, etc.) and 50 objects (books, laptops, cd-cases, etc.). Locations have been tagged with ISO15693 RFID tags, each with a storage capacity of 512 bits (each tag contains 30 slots, 1 byte each, thus it is able to store 10 pheromones). Objects have been tagged with ISO14443B RFID tags, each with a storage capacity of 176 bits (each tag contains only the object ID). In addition, we set up three HP IPAQ 36xx running Familiar Linux 0.72 and J2ME (CVM - Personal Profile). Each IPAQ is provided with a WLAN card and a M21xH RFID reader.

Moreover, a wirelessly accessible server holds a database with the associations between tag IDs and places' and objects' description (i.e. ID 001 = Prof. Smith's office door). The IPAQ can connect, via WLAN, to the database server to resolve the tag ID into the associated description. Each IPAQ runs the described agent-based application. Finally, a mobile robot has been realized by installing one of

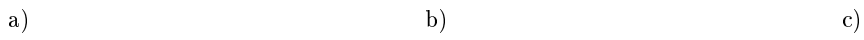

Fig. 3. (a) Our test-bed hardware implementation. (b) Some tagged objects. (c) The Lego Mindstorms robots performing pheromone search.

our wireless IPAQ (connected to a RFID reader) on a Lego Mindstorms robot. The IPAQ runs an agent controlling the RFID reader and the robot movement. This latter point has been realized by connecting, via IR, the IPAQ to the robot CPU (the RCX Lego brick) enabling the IPAQ to access robot's sensors and actuators. Robots are not provided with a localization device or map of the department. They wander randomly, avoiding collisions, looking for pheromones.

6.2 Simulation Set-Up

To test on the large scale, we realized a JAVA-based simulation of the above scenario. The simulation is based on a random graph of places (each associated to a location-tag), and on a number of objects (each associated to an object-tag) randomly deployed in the locations-graph. Each tag has been simply realized by an array of integer values. A number of agents wanders randomly across the locations-graph collecting objects, releasing objects, and spreading pheromones accordingly. At the same time, other agents look for objects in the environment eventually exploiting pheromone trails previously laid down by other agents. For the sake of comparison, we implemented 3 search algorithms: in blind-search, an agent explores the locations-graph disregarding pheromones. In local-search, the agent perceives the pheromones in its current node, but it cannot see the direction in which the pheromones increase. In grad-search, the agent perceives pheromones together with the directions in which they increase. The simulator, allows to perform a number of experiments changing a number of parameters such as the graph size, the number of objects, the number of agents involved, the storage capacity of the tags, etc. Both the real implementation and the simulation have been employed to realize the experiments described in the next subsections and to draw the conclusions described in Subsection 4.4.

6.3 Results of the Experiments

A first group of experiments (reported in Figure 3) aims at verifying the effectiveness of the application. Specifically, we set up two environments: one consisting of 100 tagged places with 100 objects (Figure 3-a) and another consisting of 2500 tagged places with 500 objects (Figure 3-b). 10 agents populate these environments wandering around moving objects and spreading pheromones and, at the same time, looking for specific objects. In the experiments, we report the number of places visited (i.e. number of location-tags perceived) before finding specific objects, for different search methods, plotted over time. These results are the average of a number (over 300) of simulated experiments and verified - on a smaller scale - on the real implementation.

The more time passes the more pheromone trails get deployed. It is easy to see that blind-search does not take advantage of pheromone trails and in fact objects are found after visiting on-average half of the places. Grad-search takes a great advantage of pheromones, in fact, after several pheromone trails have been deployed, less than 10% of the places need to be visited before finding the object. Local-search is useful only in large scenarios: the time taken wandering randomly

in a neighborhood, looking for the direction where a pheromone increases, hides pheromone benefits in small environments.

a) b)

Fig. 4. Number of places visited before finding a specific object plotted over time. (a) 100 tagged places. (b) 2500 tagged places..

A second group of experiments aims at exploring the effects of RFID tag storage saturation upon pheromone spread. This of course represents a big problem, in fact, it can happen that pheromone trails can be interrupted, because there is not available space left on neighbor location-tags, while the object to be tracked moves away. This create a broken pheromone trail leading to a place that is not the actual location of the object. In Figure 4a, we report an experiment conducted in the 100-tagged-places-environment described before. We plot the number of places visited before finding specific objects for different search methods, over a shrinking tag storage capacity. In this experiments, agents spread pheromones for 150 time steps, then they start looking for objects without spreading pheromones anymore. For sake of comparison, it is worth noting that 150 time steps is exactly the number of steps that concludes the experiment on Figure 3a. Let us focus on the grad-search method that is the most interesting in this context. It can be noticed that, when the tag storage capacity is high, we have good performance (less than 10% of the places need to be visited before finding the object). However, when the capacity fall below 85 pheromones (that is - recalling figure 1 - that the tag has a capacity of less than 85 * 3 = 255 slots = 255 bytes), performance starts decaying really fast and when the capacity is lower than 25, grad-search works equal to blind-search. It is rather easy to explain this phenomenon: when the tag capacity is low, there are a lot of broken pheromone paths degrading the performances. An agent, reaching the end of a broken pheromone trail, has no choice but starting the search from the beginning. Figure 4b shows the same problem from another perspective. This time the tag capacity has been fixed to 50 pheromones (150 bytes), and we plot

the number of places visited before finding specific objects, for different search methods, over time. Let us focus again on the grad-search behavior. It is easy to see that, when time is close to zero, grad-search works equal to blind-search, since no pheromone trails have been already laid down. After some time, grad-search works considerably better than blind-search, since pheromone trails drive agents. However, as time passes, tags capacity tend to saturate, the objects are moved, but no pheromone trails can be deployed. This situation rapidly trashes performance leading back to blind-search performance.

a) b)

Fig. 5. (a) Number of places visited before finding a specific object plotted over a shrinking tag storage. (b) Number of visited places before finding a specific object plotted over time, when tags tend to saturate

Finally, in the experiments depicted in Figure 5, we tried to assess whether the pheromone evaporation mechanism can help in such situation. Figure 5 plots the number of places visited before finding specific objects over a shrinking tag storage capacity (the same of Figure 4a). This time, however, only grad-searches are depicted and each plot is associated to a different threshold T of pheromone evaporation. Unfortunately, it is rather easy to see that the pheromone evaporation is rather ineffective.

6.4 Lessons Learnt

We get two main lessons from the experiments described in the previous section. First Lesson: in small environments grad-searches work considerably better than local-searches. However, this is not longer true in large environments, where the two methods have almost the same performance. This is clearly because the cost of "orienting" in a local neighborhood becomes negligible when the environment is large. Moreover, the drawback of grad-searches is the need for longer-range (more costly) RFID reader: the reader, in fact, must be capable of reading tags in a "one-hop" neighborhood. On the contrary local-searches can

Fig. 6. Number of places visited before finding a specific object plotted over a shrinking tag storage space, for various evaporation threshold.

work with shorter-range (cheaper) RFID reader as well. Overall, the experiments conducted show that in near-future environments (with thousands of objects and places being tagged) local-search is a promising approach. Second Lesson: the limited storage capacity of the RFID tags is a big problem. Basically, if the number of objects to be tracked is greater than the available slots on the RFID tag, in the long run the problem is unavoidable. Sooner or later, a new object will cross to an already full tag, breaking the pheromone trail. We still do not have a solution for this problem. Our research with regard to this topic is leading in two main directions: (i) we are currently researching more advanced pheromone evaporation mechanisms. (ii) We are considering the idea of spreading pheromone trails not only in location-tags but also on object-tags. The advantage would be that the more objects are in the system, the more storage space is available for pheromones, letting the system to scale naturally. The problem is how to manage the fact that object-tags containing pheromones can be moved around, breaking the pheromone trail structure. As a partial relief from this problem, it is worth reporting that, recent RFID tags have a storage capacity in the order of the KB, making possible to track thousands of objects without changing our application.

7 APPLICATIONS AND RELATED WORK

In the last few years, pheromone interaction and stigmergy have attracted more and more researches due to their power in supporting agent coordination. It is not surprising then, that both there are a number of applications - beside object tracking - that could possibly take advantage or our approach, and there are a number of related work proposing approaches to realize stigmergy either in virtual - simulated - or physical environment. In this section, we discuss these topics.

7.1 Other Application Scenarios

Other than the proposed object tracking application, our approach to actually spread virtual pheromones in the physical environment, well suits a wide range of application scenarios. Here we report two particularly broad and important examples. Context-awareness: pheromones (i.e. data structures) spread in the environment could provide context information to users. As introduced previously in the paper, localization is a simple -but important - example of this application. Our RFID implementation naturally suits this scenario, in that: (i) RFID tags can store - or link to -semantically rich contextual information, (ii) context-data can be actually spread in the environment where it will be most useful. Moreover, RFID tags stuck on objects or person could hold information of such objects and person. Reading such tags could be very valuable to asses the application context (e.g. reading the tag associated to the user boss and of a video projector can let infer that the user is in a sort of important meeting with his/her boss) [**?**]. Motion coordination: pheromones spread in the environment could enable a group of users (both humans and robotics) to coordinate their respective movements. An exemplary application would be distributed environment exploration. Users could decide to explore a specific area if there are not pheromones pointing in that direction (the area is truly unexplored). In this context, it is important to remark that our approach clearly requires the presence of RFID tags before pheromones can be spread. If the environment does not contain tags at all, our approach could not be used. However, on the one hand, RFID tags are likely to be soon densely present in everywhere (embedded in tiles, bricks, furniture, etc.). On the other hand, it is possible to conceive solutions where agents physically deploy RFID tags while exploring the environment to be used for subsequent coordination. For instance, future development in plastic (and printable) RFID technology [**?**] let us envision the possibility of enriching an agent with a simple RFID printer to dynamically print in pavements, walls, or any type of surface, RFID tags.

7.2 Related Work

In the last few years, a lot of agent applications inspired by pheromone-interaction and stigmergy have been proposed. In this section, we will report projects trying to realize real implementation of such mechanisms. In [**?**] a pheromone-based approach to coordinate Unmanned Airspace Vehicles (UAVs) is presented. This approach has been concretely implemented in a real-world scenario by spreading pheromones in a virtual data-space shared among the UAV agents. In our opinion, spreading pheromones in virtual data-space presents some drawbacks: if the data-space is centralized and globally-accessible, it creates a bottleneck for the application. If it is completely distributed among the locally interacting agents, then consistency may be hard to be maintained. In [**?**] pheromones to coordinate robots' movements are spread in a sensor network deployed over the environment. This approach is similar to our RFID implementations: agents connect to nearby sensors and store pheromones in there. In the long-term this

would be a really powerful solution: active sensors could implement pheromone evaporation autonomously, but presently it is also very costly. Also, the sensor network solution exhibit battery-exhaustion problems (and, thus, limited life) which our solution prevents. In [?] and [?] a swarm of mobile robots connect with each other in an ad-hoc network to coordinate their movements. Robots can create distributed data structures (e.g., pheromone trails) over the ad-hoc network defined by the robot themselves. In our opinion, such solution presents problems related to the cost of individual robots, and on the number of robots required to provide a good coverage of the environment and a dense enough network. Also, should the ad-hoc network of robots get partitioned, pheromone trails would be broken. A terrain-covering robot that exploits sort of pheromones is described in [?]. Here, a prototype robot is presented which is equipped with a pen to leave special ink trails in the pavement. Also, the robot is equipped with proper light sensors to sense the ink trails. In this way, robots can enforce a simple form of pheromone-based coordination (e.g., if an ink trail is sensed, it means that another robot has already covered that part of the terrain. In our opinion, the RFID tag solution is much more flexible, in that it enables using more semantic information for a wider range of applications other than terrain covering.

8 CONCLUSION AND FUTURE WORK

While a preliminary prototype implementation shows the feasibility of our approach, a number of research directions are still open to asses and improve the system practical applicability. In particular, more experiments are required to verify the scalability of the proposed architecture to hundreds of objects being possibly tracked. Moreover, effective solutions to the problem related to broken pheromone trails must be found.

9 Acknowledgements

Work supported by the Italian MIUR and CNR in the "Progetto Strategico IS-MANET, Infrastructures for Mobile ad-hoc Networks".