Intelligent Autonomous Systems 10 Baden-Baden, July 2008

Decentralized cooperative exploration: Implementation and experiments

Antonio FRANCHI [1], Luigi FREDA, Luca MARCHIONNI, Giuseppe ORIOLO and Marilena VENDITTELLI

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza", Italy

Abstract. In this paper we present implementation and experiments of a decentralized cooperative exploration strategy developed by our research group. The exploration strategy is based on the construction of a roadmap of the explored area, with the associate safe region. No task decomposition/allocation is performed. The roadmap is incrementally built through a simple and efficient decentralized cooperation mechanism: each robot biases its exploration towards its local frontier, i.e., local areas which appear to be unexplored by the whole robot team on the basis of the exchanged information. A detailed description of the software architecture used to implement the strategy is given. Experiments with a team of Khepera III robots are presented to show the performance of the proposed technique.

Keywords. Cooperative exploration, multi-robot systems, decentralized algorithms

Introduction

Exploration of unknown environments is one of the most challenging problems in robotics. This task typically requires a mobile robot to cover an unknown area while learning, at the same time, a model of the environment or locating a given object. A wide range of applications are possible including automated surveillance, search-and-rescue operations in hostile areas, map building and planetary missions.

In general, many advantages result from the use of a multi-robot system [1,2]. In exploration, a team of robots typically reduce the time required to complete the task. If a map is to be acquired, the redundant information provided by multiple robots can be also used to increase the final map accuracy and the quality of the localization [3]. In order to achieve these objectives, some sort of task decomposition and allocation are required. In practice, strategies to conveniently distribute robots over the environment should be accurately devised in order to reduce the occurrence of spatial conflicts [4] and actually reap the benefits of a multi-robot architecture. Clearly, communication plays a crucial role in achieving a cooperative behavior with improved performance [5].

In most exploration strategies, the boundary between known and unknown territory (the *frontier*) is approached in order to maximize the information gain. For the multi-

[1]Corresponding Author: Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Via Ariosto 25, 00185 Roma, Italy; E-mail: franchi@dis.uniroma1.it

robot case, a pioneering work in this direction is [6]: the robots merge the acquired information in a global gridmap of the environment, from which the frontier is extracted and used to plan the individual robot motions. While this basic scheme lacks an arbitration mechanism preventing robots from approaching the same frontier region, in [7] it is proposed to negotiate robot targets by optimizing a utility function which takes into account the information gain of a particular region, the cost of reaching it and the number of robots currently heading there. In [8], the utility of a particular frontier region from the viewpoint of relative robot localization (and hence, of the accuracy of map merging) is also considered. In the incremental deployment algorithm of [9], the robots approach the frontier while retaining visual contact with each other. An interesting multi-robot architecture in which the robots are guided through the exploration by a market economy is presented in [10], whereas [11] proposes a centralized approach which uses a frontier-based search and a bidding protocol assign frontier targets to the robots.

In this paper, we present an implementation and preliminary experiments of the SRG method, a decentralized strategy for cooperative robot exploration presented in [12]. In particular, we detail the structure of the communication threads which were not discussed in [12]. The reported experiments are conducted with a team of real robots and a detailed description of the software architecture is given.

1. Problem assumptions

For simplicity, the SRG exploration method is described under the following assumptions, which are verified for our experimental setup (see Sect. 7):

1. The robots move in a planar workspace $\mathcal{W} \quad I\!R^2$.
2. Each robot is a disk, whose configuration q is described by the cartesian position of its center. The disk is *path controllable*, i.e., it may follow any path in its configuration space with arbitrary accuracy. This assumption is verified for free-flying as well as (most) nonholonomic mobile robots.
3. Each robot is equipped with a sensory system which provides the *Local Safe Region* $\mathrm{LSR}(q)$, a (possibly conservative) estimate of the free space surrounding the robot at q within a perception range R_p.
4. Each robot can broadcast/receive data to/from other robots within a communication range R_c.

However, 3D workspaces, higher-dimensional configuration spaces and heterogeneous robots can be accommodated within the same framework. Also, path controllability can be replaced with simple controllability.

2. The Sensor-based Random Graph

The Sensor-based Random Graph (SRG) is a data structure that represents the workspace area explored by the team of robots. Nodes (arcs) of the SRG represent collision-free configurations (paths) that have been visited (traversed) by at least one robot. The Local Safe Region $\mathrm{LSR}(q)$ is also included in the description of node q (Fig. refFig:LRRLSR, right). Exploration actions are actually planned using the *Local Reachable Region* $\mathrm{LRR}(q)$, i.e.,

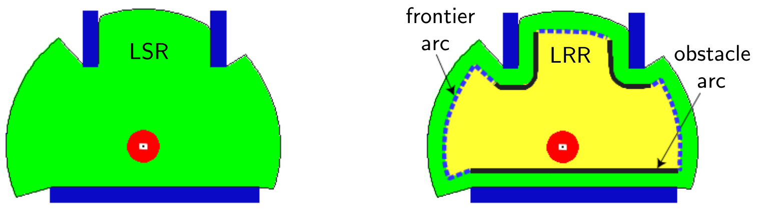

Figure 1. The Local Safe Region (left) and the Local Reachable Region (right) at a certain configuration. Also shown are the obstacle (solid) and frontier (dashed) arcs of the LRR boundary.

the set of all the configurations that can be reached from q with the robot staying within LSR(q) (see Fig. refFig:LRRLSR, right). In particular, under Assumption 2, the LRR(q) is obtained by *eroding* the LSR(q) with the robot disk as structuring element, and then *extracting* the connected component containing q.

The boundary of a LRR can be partitioned in *obstacle*, *free* and *frontier* arcs (see Fig. 1, right). The first correspond to configurations at which the robot would graze a locally detected obstacle, and are easily identified from the range scan. Free arcs do not correspond to obstacles but fall within the interior of other LRRs. Arcs that are neither obstacle nor free are frontier arcs, and their union is the *frontier* of the LRR and, by extension, of the associated node. Note that the frontier is the portion of the LRR boundary leading to unexplored areas.

Additional structures, called *bridges*, are added to the SRG to improve its connectivity. A bridge is a sequence arc-node-arc which is added to the SRG to connect two nodes v and w if the graph distance between v and w is greater than a certain threshold, and the robot can move from v to w remaining in LSR(v) [LSR(w). The first condition is aimed at

condition is aimed at improving the graph connectivity without significantly increasing its complexity, the second guarantees that the added arc is collision free. Clearly, this requires that v and w belong to the same connected component of the erosion of LSR(v) [LSR(w). The middle node of the bridge is placed inside this region, and in particular so as to fall in the intersection LSR(v) \ LSR(w). In particular, when v 2 LRR(w) (and w 2 LRR(v)), the bridge degenerates in a single arc which represents the path from v to w entirely contained in LRR(v) \ LRR(w).

The SRG is incrementally built by extending the structure in the most promising direction via a biased random mechanism. A local coordination strategy takes into account other robots in order to increase the collected information and guarantee collision avoidance.

Note that, the i-th robot of the team has its own 'representation' SRG$_i$ of the SRG, either acquired directly or through communication with other robots. The SRG cooperatively built by the team of robots is the union of all the SRG$_i$'s and is a distributed representation of the environment.

3. Functional architecture

In this section we describe the architecture of the software running on each robot of the team. In particular, we give a concise description of the function of each block in Fig. 2. Further details are given in the following sections. Blocks with solid thick boundary rep-

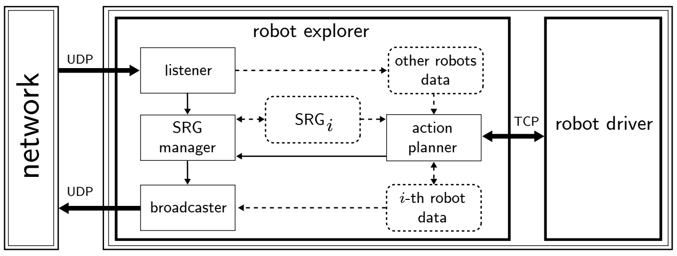

Figure 2. Functional architecture of the software running on the i-th robot.

resent processes (robot explorer and robot driver), those with thin boundary represent threads (action planner, SRG manager, broadcaster, listener), while dashed rectangles represent data (i-th robot data, team data, SRG$_i$). Arrows indicate the existence of an information flow: thick for interprocess communication, thin for communication between threads, dashed for read/write operation on data structure. The direction of the arrows corresponds to the direction of the data flow.

The *robot explorer* process implements the cooperative exploration algorithm detailed in the following section, while the *robot driver* process provides low level primitives for motion, localization and perception. The two processes communicate through the TCP protocol, allowing a distributed instantiation of the architecture and providing a flexible integrated environment for simulation and experimental validation of collaborative robot behaviors. With this architecture, in fact, the explorer and the robot driver do not need to run on the same physical machine and the robot driver can correspond to a real or to a simulated robot.

The robot explorer is realized by four threads: the *action planner*, the *SRG manager*, the *broadcaster* and the *listener*. The action planner represents the core of the robot explorer: it is in charge of choosing the exploration action in a cooperative and coordinated way. In particular, the environment information collected in the SRG$_i$ is used to choose the next view configuration in such a way that information increases. This realizes a cooperative action planning mechanism since the information stored in the SRG$_i$ has been both collected through the i-th robot driver and acquired through communication with other robots, The configurations, the planned actions and the step of the exploration algorithm currently executed by a set of robots of the team with which conflicts in the planned actions may arise are used by the action planner to actuate an appropriate coordination strategy. These data (*team data*) are made available by the listener. The same data for the i-th robot (*i-th robot data*) are provided by the robot driver (the robot configuration) and by the action planner itself and made available to the broadcaster for diffusion through the network, in order to allow the realization of the same coordination strategy by the other robots of the team. Once the next view configuration has been chosen, the action planner is also in charge of planning a safe path to it and of sending the associated motion command to the robot driver.

The task of SRG manager is to elaborate and continuously update the data stored in the SRG$_i$ on the basis of the information received from the other agents of the team and from the action planner. In particular, the listener provides the LSRs acquired by the other robots of the team with which communication has occurred, while the action planner makes available the same data acquired by the robot itself. The SRG manager merges these data so as to maintain the consistency of local representations.

The broadcaster and listener threads have dual functions: while the first propagates through the network all the information currently available to the robot, i.e., SRG_i and i-th robot data, the second collects the corresponding information from other robots of the team and makes it available to the SRG manager and to the action planner.

4. Action planner

The algorithm implemented by the action planner on the i-th robot is an instantiation of the exploration strategy presented in [12] which is based on the incremental construction of a representation of the area explored by the robots in the form of an SRG.

At the start of each iteration of the algorithm, the robot is stationary in a position corresponding to an existing node of the SRG_i. The action planner sends a perceive command to the robot driver, receives the current LSR and makes it available to the SRG manager. If the frontier of the current LRR, computed by the SRG manager, is not empty, the action planner generates a new random[2] configuration on it (see [12] for details). A path reaching the new configuration is planned inside the current LRR. Otherwise, if the frontier of the current LRR is empty, the action planner locates the nodes in SRG_i with non-empty local frontiers and plans a path toward the closest. A new iteration of the algorithm starts when the robot reaches the first adjacent node on this path.

Interlaced with perception, target selection and planning there are synchronization steps necessary to achieve coordination. Prior to choose the next view configuration, the robot has to identify the *Group of Engaged Agents* (GEA), i.e, the other agents of the team with which cooperation and coordination are necessary, based on the information stored in its SRG_i and on the other robots data. Formally, two robots are *GEA-coupled* when their LSRs overlap. The GEA of the robot is composed by all the robots to which it can be connected through a chain of GEA couplings. Since the other robots of the team may be stationary as well or moving, a synchronization phase is needed. To this end, the robot first builds the *Group of Pre-engaged Agents* (GPA), i.e., the robots which are candidate to belong to the GEA. Two robots are *GPA-coupled* if the distance between their targets is at most $2R_p$. The GPA of the robot is formed by all the robots to which it can be connected through a chain of GPA couplings. To achieve synchronization, the GPA is computed and updated with the data made available by the listener until all its members are stationary.

The actual GEA is built when the LSRs of all the robots in its GPA have been received by the listener and integrated in SRG_i by the SRG manager. If the GEA is composed only by the robot itself, the action planner sends a motion command to the robot driver and the robot moves to its target. Otherwise, it waits for receiving all the prospective paths of the robots in the GEA and checks them for mutual collisions. Collision paths are then classified in feasible and unfeasible according to a deterministic rule implemented by each robot. If the planned path is not feasible, the robot's move is forbidden by resetting the target to its current configuration. Last, a motion command leading the robot towards its target is sent to the robot driver.

[2] A deterministic version of this planner can be envisaged in which a specific configuration on the closer frontier is selected according to some criterion. However, given the large branching factor of the exploration problem, a biased random procedure can be considered a valid (and easy-to-implement) alternative to more computationally expensive greedy strategies.

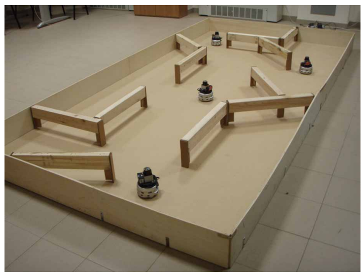
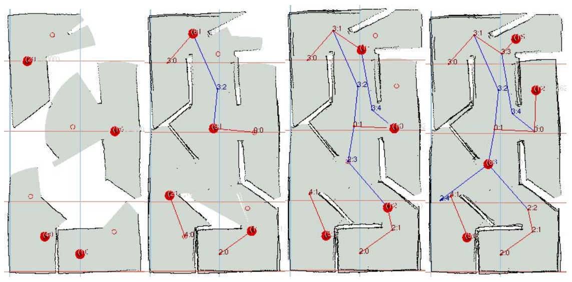

Figure 3. *Left*: the first environment. *Right*: an exploration progress in the first environment, as reconstructed by the visualizer. Bridges are depicted in blue.

Exploration iterations are repeated until there are no reachable nodes in the SRG$_i$ with non-empty frontier. This means that the boundary of the explored region reachable by the robot is only composed by obstacle arcs. Hence, the robot is unable to further expand the graph and it has to go back to its starting position (*homing*).

5. SRG manager

The SRG manager receives data from the action planner and the listener and updates SRG$_i$ accordingly. Two kind of data can be received: a node or an arc. The node comes with its associated LSR(q), while the arc consists in a couple of nodes to be joined by a feasible path. On reception of a node (either from the action planner or from the listener), the SRG manager 1) adds the node to SRG$_i$ or updates the info associated to the preexisting node 2) computes the LRR(q) and the associated frontier 3) updates the frontiers of the nodes in SRG$_i$ whose LRRs have a non-empty intersection with LRR(q). On reception of an arc, new nodes[3] are added to the SRG$_i$ if they are not already stored in the graph. Then, the new arc is attached to the two nodes. Each time a new node v is added to SRG$_i$, the SRG manager creates a bridge between v and any node w in SRG$_i$ satisfying the two conditions given in Sect. 2.

6. Broadcaster and listener

On each robot, the broadcaster and the listener respectively transmit and receive information. This is organized in three possible messages containing 1) the current robot configuration, the current target, the step of the exploration algorithm currently executed by the action planner, the robot's GPA/GEA[4]; 2) the nodes between which an arc is to be created; 3) the node, either new or already present in the SRG$_i$, with the associated LSR. Messages of the first type are transmitted synchronously by the broadcaster, while the other two kind of messages are transmitted as new data are made available from the SRG manager. The listener receives messages asynchronously from the network and makes them available to the action planner (other robots data) and the SRG manager.

[3]To reduce bandwidth consumption, the nodes do not come with an associated LSR.
[4]This information is necessary only if the communication range R_c is limited.

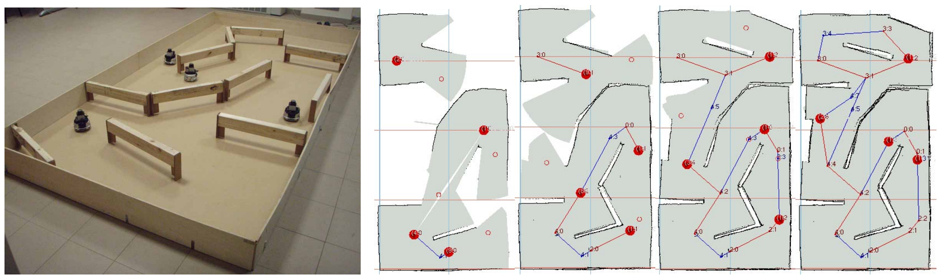

Figure 4. *Left*: the second environment. *Right*: an exploration progress in the second environment, as reconstructed by the visualizer. Bridges are depicted in blue.

	first experiment							second experiment						
	robots				aggregate data			robots				aggregate data		
	1	*2*	*3*	*4*	*mean*	*st dev*	*total*	*1*	*2*	*3*	*4*	*mean*	*st dev*	*total*
# nodes	5	7	2	3	4	2	17	4	5	7	3	5	2	19
# total arcs	5	7	1	5	5	3	18	4	5	9	3	5	3	21
# bridge arcs	3	4	0	0	2	2	7	2	2	6	2	3	2	12
traveled distance (m)	3,34	2,65	1,28	2,35	2,41	0,86	9,62	3,42	4,05	3,98	1,27	3,18	1,30	12,72
exploration time (sec)	240	228	92	174	184	67	240	265	259	264	179	242	42	265
homing error (m)	0,053	0,055	0,036	0,022	0,042	0,016	0,166	0,020	0,074	0,018	0,156	0,067	0,065	0,268

Figure 5. Table of numerical results.

7. Experiments

A preliminary simulation study in [12], performed with teams of various cardinality, has shown that both the exploration time and the mean traveled distance quickly decrease in the beginning with the number of robots, then asymptotically tend to a constant value. In the case of scattered start this asymptotic value is zero while in the case of clustered start it depends on the size of the environment. In the present work, experiments have been conducted with a team of 4 Khepera III robots (http://www.k-team.com). In addition to the standard equipment, each robot has been provided with a wireless card, for communication between robots of the team and/or with a remote computer, and a Hokuyo URG-04LX laser rangefinder, used for the construction of the LSR, with the following characteristics: angular resolution of 0.36 , radial resolution of 1 mm, scanning angle equal to 240 , maximum perception range of 4 m. In the adopted experimental setup, a robot driver runs on each Khepera III, whereas each explorer process can flexibly run on any separate remote computer. During the exploration, an additional process, called *visualizer*, is in charge of 'sniffing' and storing all the packets exchanged among explorer processes in order to visualize and monitor the exploration progress. Although all the robot explorers and robot drivers communicate with each other through a single LAN, hence sharing the same bandwidth, the limited set of exchanged messges prevented any relevant loss of communication packets in all the conducted experiments.

Each robot is provided with a basic self-localization module in which incremental scan matching [13] is used to correct odometry localization. A preliminary calibration of robot odometry and sensor parameters was performed beforehand as described in [14]. In these preliminary experiments, the robots know their relative configurations at the start of exploration. Hence, mutual localization is maintained on the basis of this initial knowledge and the above mentioned local registrations.

Figures 3 and 4 show two typical exploration experiments. The depicted environments (Fig. 3 and Fig. 4, left) have been constructed in a rectangular arena whose area is approximately 8 m^2. In these experiments, for each robot, the laser perception range has been limited to 1 m and the maximum linear velocity was 15 cm/sec. The robot communication range was not artificially limited[5]. At the end of each experiment, the robots return back to their starting positions (homing). Figures 3 and 4, right, show the exploration progress as reconstructed by the visualizer. The relevant numerical data associated to the experiments are reported in Fig. 5.

Note that the environment considered in the first experiment is simply connected and its topology is correctly captured by the resulting SRG in the form of a tree (see Fig. 3, right). Analogously, the multiply connected environment of the second experiment is efficiently represented by the resulting SRG (see Fig. 4, right). It is worth noting that, in this case, the number of bridges is higher than in the first experiment in order to better accommodate the presence of loops. Additional experiments, including movies, are available at `http://www.dis.uniroma1.it/labrob/research/multiSRG.html`.

8. Conclusions

In this paper we have presented the implementation and experiments of the SRG method, a decentralized cooperative exploration strategy for mobile robots. Future work include: design of a procedure for global alignment of scans, based on the work in [15], to be performed each time a loop closure is detected; quantitative study of the robustness and scalability properties of the method; development of a mutual localization method.

References

[1] Y. Cao, A. Fukunaga, and A. Kahng, "Cooperative mobile robotics: Antecedents and directions," *Autonomous Robots*, vol. 4, no. 1, pp. 7–27, 1997.

[2] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes, "A taxonomy for multi-agent robotics," *Autonomous Robots*, vol. 3, pp. 375–397, 1996.

[3] I. Rekletis, G. Dudek, and E. Milios, "Multi-robot collaboration for robust exploration," *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1, pp. 7–40, 2001.

[4] D. Goldberg and M. Mataric, "Interference as a tool for designing and evaluating multi-robot controllers," in *14th AAAI/9th IAAI*, 1997, pp. 637–642.

[5] T. Balch and R. Arkin, "Communication in reactive multiagent robotic systems," *Autonomous Robots*, vol. 1, no. 1, pp. 27–52, 1994.

[6] B. Yamauchi, "Frontier-based exploration using multiple robots," in *2nd Int. Conf. on Autonomous Agents*, 1998, pp. 47–53.

[7] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. on Robotics and Automation*, vol. 1, no. 3, pp. 376–386, 2005.

[8] J. Ko, B. Stewart, D. Fox, K. Konolige, and B. Limketkai, "A practical, decision-theoretic approach to multi-robot mapping and exploration," in *2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003, pp. 3232–3238.

[9] A. Howard, M. Mataric, and S. Sukhatme, "An incremental deployment algorithm for mobile robot teams," in *2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003, pp. 2849–2854.

[5]Note that one of the main features of the SRG method is that it does *not* require an unlimited communication range. See [12] for further details.

[10] R. Zlot, A. Stenz, M. Dias, and S. Thayer, “Multi-robot exploration controlled by a market economy,” in *2002 IEEE Int. Conf. on Robotics and Automation*, 2002, pp. 3016–3023.

[11] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes, “Coordination for multi-robot exploration and mapping,” in *17th AAAI/12th IAAI*, 2000, pp. 852–858.

[12] A. Franchi, L. Freda, G. Oriolo, and M. Vendittelli, “A decentralized strategy for cooperative robot exploration,” in *First Int. Conf. on Robot Communication and Coordination*, 2007.

[13] A. Censi, “An icp variant using a point-to-line metric,” in *2008 IEEE Int. Conf. on Robotics and Automation*, 2008.

[14] A. Censi, L. Marchionni, and G. Oriolo, “Simultaneous maximum-likelihood calibration of robot and sensor parameters,” in *2008 IEEE Int. Conf. on Robotics and Automation*, 2008.

[15] F. Lu and E. Milios, “Globally consistent range scan alignment for environment mapping,” *Autonomous Robots*, vol. 4, pp. 333–349, 1997.