Robotics and Autonomous Systems 58 (2010) 529–538

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Energy constrained multi-robot sensor-based coverage path planning using capacitated arc routing approach

Aydin Sipahioglu[a,*], Gokhan Kirlik[a], Osman Parlaktuna[b], Ahmet Yazici[c]

[a] Industrial Engineering Department, Eskisehir Osmangazi University, Eskisehir, 26480, Turkey
[b] Electrical Engineering Department, Eskisehir Osmangazi University, Eskisehir, 26480, Turkey
[c] Computer Engineering Department, Eskisehir Osmangazi University, Eskisehir, 26480, Turkey

ARTICLE INFO

Article history:
Received 21 June 2009
Received in revised form
13 January 2010
Accepted 15 January 2010
Available online 21 January 2010

Keywords:
Multi-robot
Sensor-based coverage
Capacitated arc routing problem

ABSTRACT

Multi-robot sensor-based coverage path planning requires every point given in the workspace has to be covered at least by a sensor of a robot in the robot team. In this study, a novel algorithm was proposed for the sensor-based coverage of narrow environments by considering energy capacities of the robots. For this purpose, the environment was modeled by a Generalized Voronoi diagram-based graph to guarantee complete sensor-based coverage. Then, depending on the required arc set, a complete coverage route was created by using the Chinese Postman Problem or the Rural Postman Problem, and this route was partitioned among robots by considering energy capacities. Route partitioning was realized by modifying the Ulusoy partitioning algorithm which has polynomial complexity. This modification handles two different energy consumptions of mobile robots during sensor-based coverage, which was not considered before. The developed algorithm was coded in C + + and implemented on P3-DX mobile robots both in laboratory and in MobileSim simulation environments. It was shown that the convenient routes for energy constrained multi-robots could be generated by using the proposed algorithm in less than 1 s.

1. Introduction

Complete coverage path planning is the determination of a path such that every point in a given workspace is covered at least once [1]. It is required in a variety of applications such as vacuum cleaning robots [2] and lawn mowers [3]. In these applications robot apparatus is expected to move over all the points. In some other applications, such as patrolling and search–rescue operations [4], it is enough to move the robot so that sensors on the robot cover all the points of the environment. In these applications, robot has not to pass over all the points of the environment. This type of coverage is referred as sensor-based coverage [5].

In sensor-based coverage of narrow or cluttered spaces, where obstacles lie within the detector range, Generalized Voronoi Diagram (GVD) [6] can be used to model the environment. For complete sensor-based coverage of the given area, it is sufficient for the robot to follow the GVD [5]. Additionally, a representative graph of the environment could be constructed based on GVD. Then, a better time-effective coverage path can be obtained using this graph and some specialized algorithms [7]. Using multiple robots may reduce the time required to complete the coverage task and enhances robustness compared to the single robot. There are some approaches for multi-robot coverage path planning [8–10] in the literature. In [8], the cost was evaluated in terms of traveled distance by the robot. They used arcs of the configuration space and Voronoi diagram to compute the paths in the whole area. Firstly, a tour was generated for traversing all the paths, and then appropriate parts of the tour were assigned to each robot according to a distance-based cost evaluation. However, the method of partitioning the path among the robots considering their energy capacity was not considered. The approach in [9] presented an adaptation of the single-robot cellular decomposition approach to multiple robots. A robot team moves in formation to cover cells. As cells are created and/or completed, the team splits up into smaller teams, each of which continues coverage task. In that work also, the energy capacities of the robots were not considered. In another work [10], mobile robot deployment problem was considered for a specific type of coverage problem. The deployment problem was described as determination of the number of groups unloaded by a carrier, the number of robots in each group, and the initial locations of those robots. Both timing and energy constraints of robots were considered. This work only considers simple rectangular scanlines as the coverage routes, and solves the deployment problem using fewer robots in vast environments. However, the deployment was not considered for narrow-complex environments. So, to the best of the authors' knowledge, there is no work in the literature that considers efficient multi-robot sensor-based coverage of narrow environments considering robot energy capacities.

* Corresponding author. Tel.: +90 222 2303972; fax: +90 222 2213918.
E-mail addresses: asipahi@ogu.edu.tr (A. Sipahioglu), gokhankirlik@gmail.com (G. Kirlik), oparlak@ogu.edu.tr (O. Parlaktuna), ayazici@ogu.edu.tr (A. Yazici).

0921-8890/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2010.01.005

In this study, multi-robot sensor-based coverage of narrow environments is achieved considering the robot energy capacities. Firstly, a GVD-based model of the environment is constructed, and then the sensor-based coverage problem is described as partitioning arcs of the GVD-based graph among the robots without exceeding their energy capacities [11]. This problem resembles the capacitated arc routing problem (CARP). The CARP is a problem that aims to construct tours for vehicles to minimize total traverse without exceeding energy capacity of the vehicles [12]. The vehicle capacity in the CARP correspond the energy capacity of a robot in the coverage problem. In the CARP, amount of the material to be collected (or distributed) on an arc is defined as the arc demand. However, in the coverage problem, two different arc demands based on the robot's energy consumption could be defined. The robot passes through an arc on the GVD with or without covering. These situations require different energy consumption for a given arc. This is the difference between the CARP and the coverage problem. Due to this difference, classical CARP solution techniques cannot be used directly for the coverage problem. Therefore, a new solution technique or modification of an existing method is required. In this study, the Ulusoy partitioning heuristic was modified and used to solve the multi-robot sensor-based coverage path planning problem (MRSBCP) considering robot energy capacities. The Ulusoy partitioning algorithm is chosen because of its flexible nature for modification and ability to reach a near optimal solution in a short time.

The rest of this paper is arranged as follows. In Section 2, the CARP and its relation with sensor-based coverage are explained. In Section 3, the proposed robot control architecture is given. Experimental results both using real robots and simulation environment are given in Section 4. Conclusions and discussions are given in the final section.

2. Capacitated arc routing problem and relation with sensor-based coverage

A graph can be modeled as $G(V, A)$ where V is the set of vertices and A is the set of arcs connecting the vertices. A graph is called undirected if all arcs have no direction. Otherwise, it is called directed. If it is possible to reach all of the vertices through existing arcs, the graph is called connected, otherwise disconnected [13]. In addition, if all the vertices have a connection with other vertices directly, the graph is called complete, otherwise sparse. A weighted graph is one where distance or cost is assigned to its arcs. The weight of the arc can be calculated as the Euclidian distance between each pair of vertices or traveling cost. The matrix of Euclidian distances is the distance matrix. In a sparse and connected graph, a shortest path algorithm should be used to form the distance matrix and there are numerous algorithms to find the shortest path between any two vertices such as the Floyd algorithm [13]. In a complete and planar graph there is no need to calculate the shortest distance.

A tour is defined as Eulerian if it is possible to return to the starting vertex by passing through each arc exactly once [14]. If an Euler tour does not exist, some arcs must be visited twice or more to return the starting point. In this case, it is important to determine the shortest tour and the Chinese Postman Problem (CPP) occurs. The CPP is defined as determination a tour of minimum cost or minimum length that visits each arc at least once. Since the CPP is not an NP-hard problem, both mathematical model and heuristics such as Edmonds and Johnson's Minimum Perfect Matching algorithm [14] and the Hierholzer algorithm [13] can be used to obtain the CPP tour. In this problem, if some arcs of the graph need to be visited, the CPP turns into the Rural Postman problem (RPP). Unlike the CPP, the RPP is an NP-hard problem [15]. In the RPP, the arcs are divided into two groups as requiring service (required arc set) and not requiring service (non-required arc set). If the graph of the required arc set is connected, the RPP tour can be obtained like finding the CPP tour by using the algorithms mentioned above with an optimal solution. However, the graph of the required arc set may be disconnected. In this case, the Frederickson heuristic can be used to determine the RPP tour. In addition, the RPP tour may be unnecessarily long when it is constructed by using only the required arcs. In this case, the tour may be improved by using the Shorten Algorithm [16].

If there are more than one vehicle having certain capacities and the aim is to obtain the minimum total distance without exceeding capacity constraints, the problem is called the capacitated arc routing problem (CARP). The CARP was originally proposed by Golden and Wong, and was proven to be NP-hard [17]. Exact solution methods were proposed for the CARP, such as hierarchical relaxations lower bound of Voß and Amberg [18], valid inequalities of Belenguer and Benavent [19], and cutting-plane-based algorithm [20]. Recently, Wøhlk has developed new lower bounds for the classical CARP [21]. Since obtaining the optimal solution is not easy due to the NP-hard nature of the problem, different heuristic algorithms have been developed in the literature such as simple constructive methods, and two-phase constructive methods [12]. Additionally, meta-heuristic algorithms were also developed such as tabu search-based algorithm [22].

Although there are many solution techniques to solve the CARP, these techniques cannot be directly applied to the MRSBCP. In this problem, there are more than one robot (vehicle) having certain capacities. Since arcs are defined as required and non-required, and the robots have limited energy capacity, this problem resembles the CARP. However, unlike the CARP, there are two types of arc demand which can be called traveling energy and task performing energy (coverage energy) in the MRSBCP. When the robot passes through a non-required arc, there is no need to consume coverage energy for this arc and the arc demand is determined by only the traveling energy which arises from motors, embedded computer, microcontroller card, and navigation sensors (sonar) [10]. On the other hand, for a required arc, the robot consumes energy for both traveling and performing its coverage task. Namely, robot consumes additional energy for covering the environment by its coverage sensor (for example camera, laser range finder, thermal sensor, etc.). Then, the arc demand is calculated by adding the traveling energy and the coverage energy. Defining two different demands for an arc is quite different from the classical CARP. Because of this critical difference, the CARP solution techniques (both exact methods and heuristics) cannot be used directly for the MRSBCP. Therefore, a new solution technique or modification of an existing method is required.

In this study, a well-known CARP heuristic, the Ulusoy partitioning algorithm [23], was modified and used to solve the MRSBCP. The Ulusoy partitioning algorithm is chosen due to following reasons. Firstly, it is a constructive heuristic and generates a feasible solution in a polynomial time. Secondly and more importantly, this algorithm is flexible to adapt to the MRSBCP. In the CARP literature, there are many heuristic algorithms to generate a feasible solution for the CARP. Most of these heuristics generate the tour by connecting the current location of the vehicle to the depot when the total load exceeds the capacity of the vehicle. However, this approach is not suitable for the MRSBCP, because the robot will consume traveling energy to return to the depot and addition of this energy may result consuming the whole energy of the robot before reaching the depot. The Ulusoy partitioning algorithm permits to control the total energy requirement of a constructed tour. Therefore, a modified version of the Ulusoy partitioning algorithm is used for the MRSBCP.

The Ulusoy algorithm needs an Euler tour at the beginning that can be constructed using the CPP or RPP solution approaches depending on the required arc set. Then the algorithm constructs an intermediate graph by considering vehicle capacities. Later, a shortest path algorithm is used to determine both the minimum number of required vehicles to visit all the arcs and tour for each vehicle. In the following paragraphs, the modified Ulusoy algorithm (MUA) is given in detail.

Parameters used in the proposed algorithm are:
e_{ij}: Energy consumption during the travel through the arc (v_i, v_j) due to motors, microcontroller units, etc., and
q_{ij}: required energy for sensor coverage and traveling of the arc (v_i, v_j).
R: The required arc set.

Initially each robot has limited energy capacity (E_{cap}) and must return to the home location (depot) before consuming its full energy.

Initialization step (Constructing the Euler tour)

i. Create a graph $G = (V, A)$ of the environment based on the GVD. Get the required arc set R.
ii. Get all the information about the robots: Number of available robots m, the energy capacity (E_{cap}) of the robots.
iii. If there exist any unidirectional or closed arc between any pair of vertices, use the Floyd's algorithm to find the shortest path between that pair of vertices. Using these shortest paths, update the distance matrix D such that all $d_{ij} > 0$.
iv. Using the proposed approach given in [7], find the Euler tour T for the required arc set (R) for a single robot with infinite energy capacity.

The modified Ulusoy algorithm:

Step 1: Re-label the vertices in G so that the given tour T is equal to $(v_0, v_1, \dots, v_t = v_0)$, where v_0 is the depot. Let r be the largest index of a vertex incident to a serviced arc on T. Construct a directed graph $G' = (V', A)$ with vertex set $V' = \{v_0, v_1, \dots, v_r\}$ and introduce arc pairs (v_a, v_b) for $a, b = 1, 2, 3, \dots, r$ in A that satisfy $b > a$. Remove all arcs (v_a, v_b) for $a, b = 1, 2, 3, \dots, r$ such that $b > a + 1$ and (v_a, v_{a+1}) or (v_{b-1}, v_b) is not a serviced arc on T. Define C_{ab} as the cycle between v_a and v_b, and d'_{ab} as the distance cost of C_{ab} in G. Consider different cases as follows.

- If $b = a + 1$ and (v_a, v_{a+1}) is not a required arc on T, then set $d'_{ab} = 0$.
- If $b > a + 1$ or (v_a, v_{a+1}) is a required arc.
  - If chain P_{ab} contains the depot on T. $P_{ab} = (v_a, \dots, \text{depot}, \dots, v_b)$ (Depot could be anywhere in the chain P_{ab}). Then add to P_{ab} the shortest chain between v_a and v_b in G. Then, the cycle $C_{ab} = (\text{depot}, \dots, v_a, SP_{ab}, v_b, \dots, \text{depot})$ is obtained. Here SP_{ab} represents the shortest path between v_a and v_b.
  - If chain P_{ab} does not contain the depot on T. $P_{ab} = (v_a, \dots, v_b)$. Then add to P_{ab} the shortest chain between the depot and v_a, and the shortest chain between the depot and v_b in G. Then, the cycle $C_{ab} = (SP_{\text{depot } a}, v_a, \dots, v_b, SP_{\text{depot } b})$ is obtained.
  - Calculate the total energy load for C_{ab} using the non-required arc demand e_{ij} and the required arc demand q_{ij} for all arcs in C_{ab}.
    - ■ If the total load does not exceed E_{cap}, calculate distance cost d'_{ab} of (v_a, v_b) in G', defined as the total distance cost of C_{ab} in G.
    - ■ Else remove (v_a, v_b) from G'.

Step 2: Solve a shortest path problem from v_0 to v_r in G'. Each arc (v_a, v_b) used in the shortest distance path corresponds to a feasible vehicle route on G.

To explain the procedure and notation of the algorithm in detail, the following illustrative example is given. A sample graph G and corresponding RPP tour on G that covers all required arcs (R) is given in Fig. 1. The input parameters for this example of the MUA are given below.

T : 1–2–3–4–5–1–4–1
R : (1–2), (1–4), (1–5), (3–4), (4–5) (solid lines in G)
$E_{\text{cap}} = 10$.

Distances between vertices ($d_{ij}, \forall(i, j) \in E$) are shown on G.

$e_{ij} = 1, \quad \forall(i, j) \in E$ (Traversing energy)
$q_{ij} = d_{ij}, \quad \forall(i, j) \in R$ (Coverage energy).

Firstly, for the input tour, the algorithm labels all vertices on tour T by v_1 through v_8, then determines the index r which represents the largest index of a vertex incident to a serviced arc on T, $r = v_7$. Afterwards, directed graph G' is constructed with vertices v_1 through v_7, and the arc set between (v_i, v_j) where $i < j$ for $i, j = 1, 2, \dots, 7$. Since (v_2, v_3) is a non-serviced arc on G, its cost is set to 0 in G'. Using the "Remove all arcs (v_a, v_b) for $a, b = 1, 2, 3, \dots, r$ such that $b > a + 1$ and (v_a, v_{a+1}) or (v_{b-1}, v_b) is not a serviced arc on T" step of the algorithm, arcs $(v_1, v_3)\{(v_a, v_{a+1})$ is a non-serviced arc$\}$ and (v_2, v_4), (v_2, v_5), (v_2, v_6), (v_2, v_7) arcs $\{(v_{b-1}, v_b)$ is a non-serviced arc$\}$ are removed from G', because, (v_2, v_3) is not a serviced arc on T. Besides, arcs (v_1, v_6), (v_1, v_7), and (v_3, v_7) are removed due to the capacity violation. The resultant directed graph G' which shows feasible robot tours is given in Fig. 2. Finally, a shortest path between the source and the terminal vertices is obtained on G' to determine minimum cost robot tours that covers all required arcs which is given in Fig. 3.

The obtained shortest path shows that the tour must be partitioned into two different tours and two robots is enough to cover all required arcs. The robot tours are given in Fig. 4. The first robot covers arcs (1–2) and (3–4), and the second robot covers arcs (1–4), (4–5), and (1–5). In Fig. 4, the covered arcs are shown by solid lines whereas deadheading arcs are shown by dashed lines.

The MUA finds a feasible solution when there is enough number of robots with appropriate energy capacities. However, since it is a heuristic algorithm, it does not guarantee an optimal solution. Additionally, the objective function of this algorithm is to minimize the total tour length, the solution is distance near optimal. If an objective function is used to minimize total consumed energy, the solution would be energy near optimal.

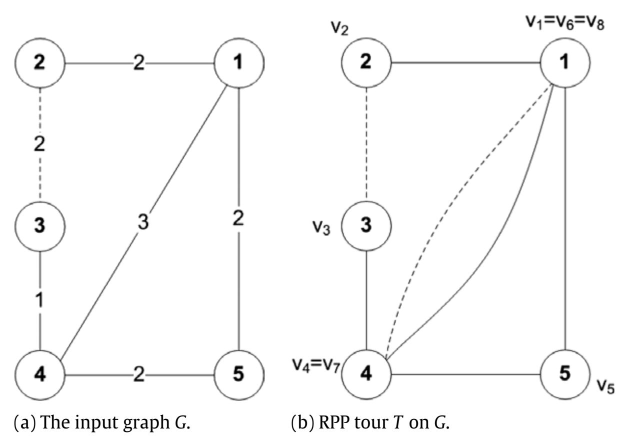


Fig. 1. Sample graph and corresponding RPP tour for the illustrative example.

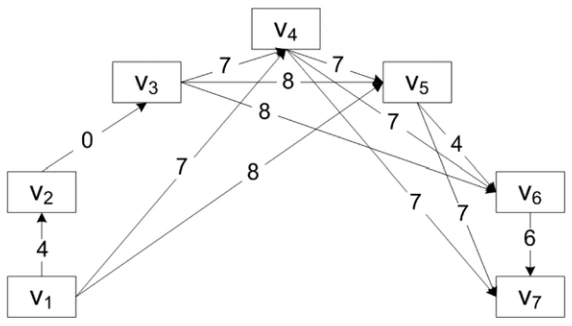


Fig. 2. Directed graph G' that represents feasible robot tours.

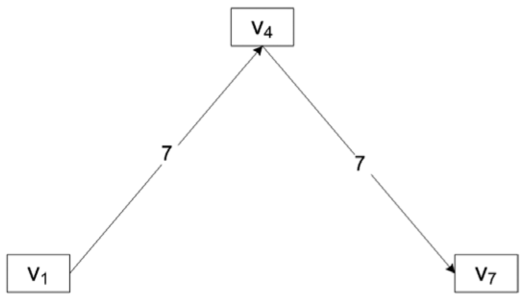


Fig. 3. Shortest path between vertices v_1 and v_7 in G'.

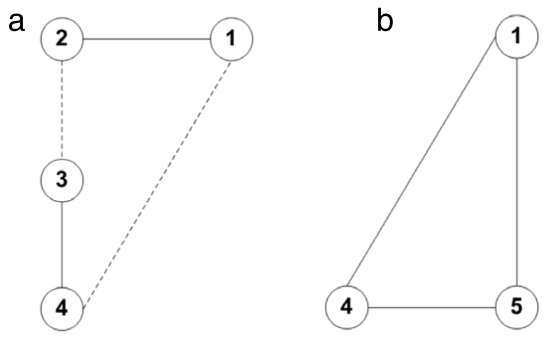


Fig. 4. Shortest path between vertices v_1 and v_7 in G'.

In this study, the MUA is used to generate path plans for each robot in the team. This algorithm is modified to handle the traveling demands in the MRSBCP. Algorithmically, the main difference between the MUA and the Ulusoy partitioning algorithm is in calculation of energy consumption. Total servicing demand on tour partition can be calculated before the tour construction. However, total traveling demand cannot be determined, unless the tour is constructed. In the Ulusoy partitioning algorithm, energy consumption for each giant tour partition is calculated before the tour construction for the robot by using the total load on tour partition. Therefore, the Ulusoy partitioning algorithm cannot be used. On the other hand, in the MUA, energy consumption on the giant tour partition is calculated after the tour construction.

The MUA generates tours according to the given robot energy capacity. This information would also be interpreted as how many robots are needed to cover the environment with the given robot energy capacity. But, in some cases number of robots could be restricted. In this case, the MUA's solution requires more robots than the available number of robots. In such a situation, either additional robots should be used for complete coverage or some regions in the environment may not be covered due to the restriction on number of available robots.

3. Robot control architecture integrated with the proposed planning approach

Mobile robot applications require coordination of several modules such as perception, localization, motion planning, low-level control, human robot interface etc. Depending on the application, different robot control architectures are used to coordinate these modules. In this study, agent-based control architecture is proposed for multi-robot sensor-based coverage path planning and execution. The architecture consists of software agents performing specific tasks required for the overall control of the robot team as seen in Fig. 5. In the system, there are *n* homogeneous robots. Each dashed square in Fig. 5 represents a robot in the team, and each block inside a robot represents a software agent. Each software agent in the robot has a different functionality. These functionalities may be communication with other robots, perception, action, negotiation, planning, human interaction, etc.

In the proposed architecture, each robot has four functional software agents: user interface agent (*UIA*), planning agent (*PA*), action agent (*AA*), and communication agent (*CA*). All of the agents are separate computational processes which perform their tasks according to their designed purposes. While the agents carrying out their regular tasks, they also response to messages coming from other agents. To facilitate messaging between the agents, Open Agent Architecture (OAA) framework is used [24]. In this framework, a facilitator agent provides content-based message transfer which enables the agents to receive the interested messages. The functional details of the agents are given as follows.

The User Interface Agent (*UIA*): The main task of the agent is to provide interaction between the users and robots. It provides a

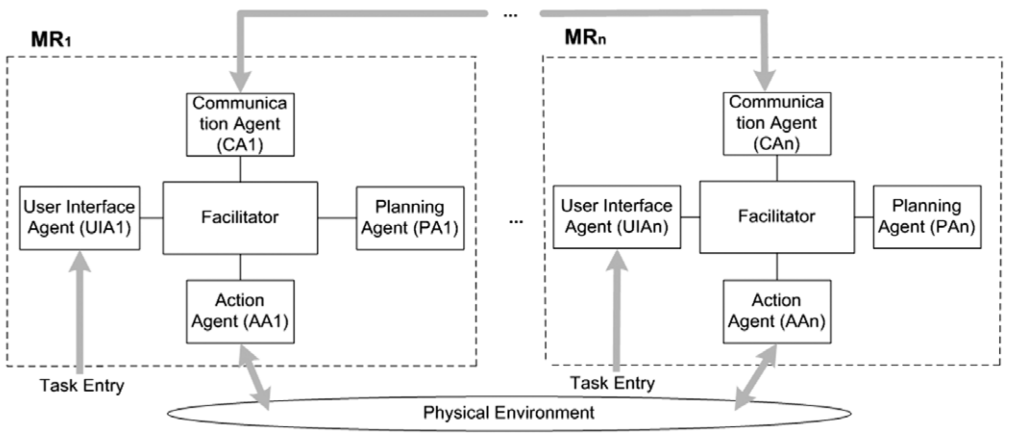


Fig. 5. Overall block diagram of the architecture.

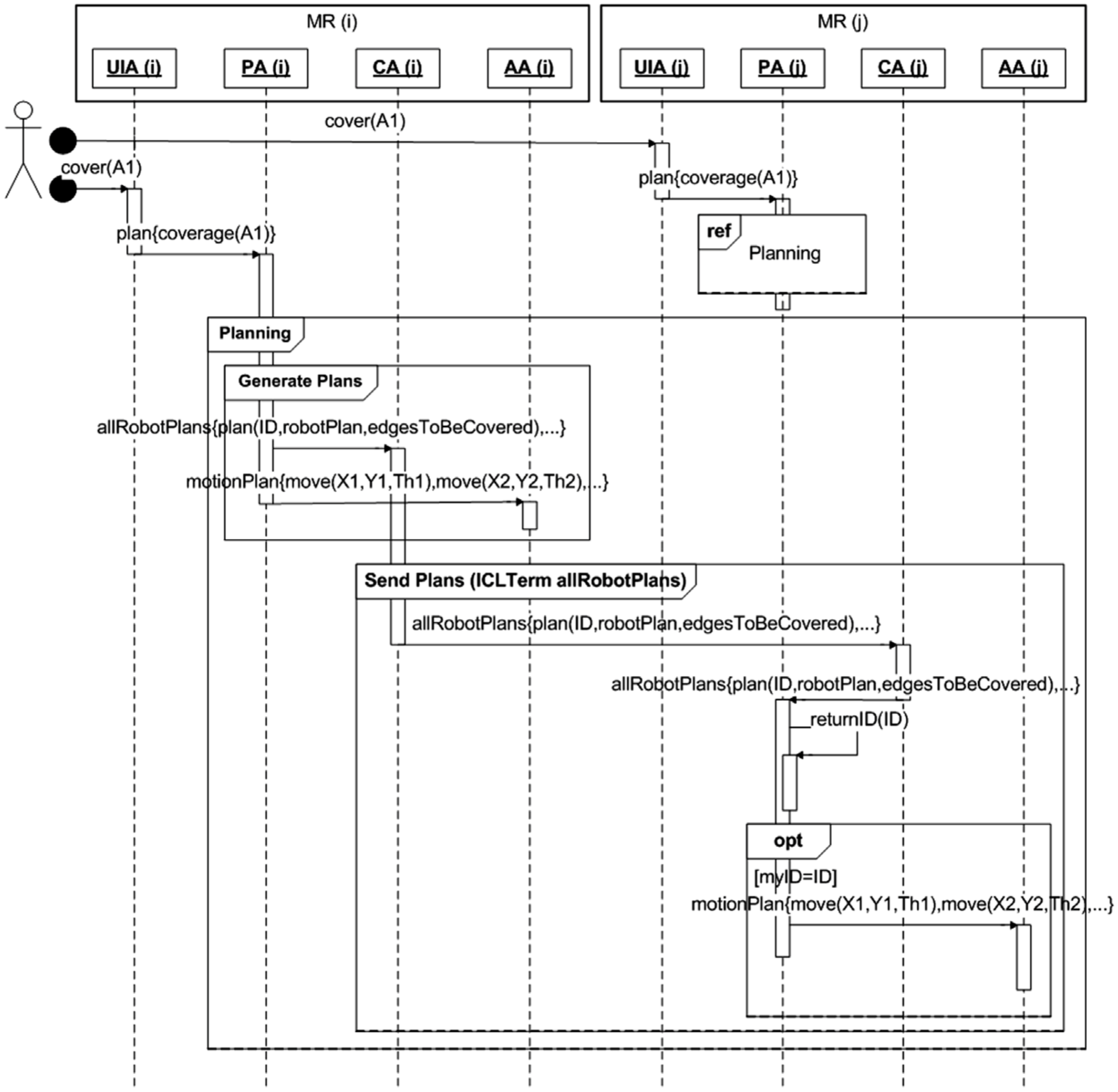

MR (i)
UIA (i)
PA (i)
CA (i)
AA (i)
MR (j)
UIA (j)
PA (j)
CA (j)
AA (j)
cover(A1)
cover(A1)
plan{coverage(A1)}
plan{coverage(A1)}
ref
Planning
Planning
Generate Plans
allRobotPlans{plan(ID,robotPlan,edgesToBeCovered),...}
motionPlan{move(X1,Y1,Th1),move(X2,Y2,Th2),...}
Send Plans (ICLTerm allRobotPlans)
allRobotPlans{plan(ID,robotPlan,edgesToBeCovered),...}
allRobotPlans{plan(ID,robotPlan,edgesToBeCovered),...}
returnID(ID)
opt
[myID=ID]
motionPlan{move(X1,Y1,Th1),move(X2,Y2,Th2),...}

Fig. 6. Sequence diagram of planning.

user-friendly environment to define tasks and provide information about internal state of the robot via terminal or graphical interface.

The Planning Agent (*PA*): This agent is responsible for generating plans which may include motion planning, path planning, task planning, etc. It contains planning mechanisms which use the planning algorithms for different task domains. It accepts tasks from the *UIA* and generates plans. The plans may be generated for one or more robots. It is assumed that all robots have a GVD-based representative network (named as A_i) of the environment. Traversing all arcs of the network guarantees full sensor-based coverage. This is also the agent where our proposed coverage planning algorithm is embedded in.

The Action Agent (*AA*): The action agent realizes perception, localization, and action at the same time because of constraints in the Aria library run on Pioneer 3-DX robots. This agent is responsible for tasks related to hardware of robots (actuators and sensors). It performs behavior-based motions according to motion plans provided by the *PA*. The task-oriented behaviors are implemented in conjunction with the survival behaviors like obstacle avoidance by coordination of subsumption [25] approach. Meanwhile, this agent is also responsible for the robot's high-level perception and localization. Localization problem is solved using ARNL's [26] laser-based localization module in the Aria software architecture. This composition also reduces the communication costs of the modules. After the action agent receives the motion plan, it starts to follow the path described by the motion plan.

The Communication Agent (*CA*): This agent is responsible for interaction with the other robots. The interaction takes place over the infrastructure realized using TCP/IP. Robots may communicate each other explicitly via the *CA*.

These agents interact with each other as in the sequence diagram in Fig. 6. In the proposed robot control architecture, the coverage task is initiated by using any robot's *UIA* in the team. User enters the mission input to the MR_i's user interface agent (UIA_i) like "*cover*($A1$)" via the user terminal. Here, "*cover*" shows the mission and "$A1$"declares the coverage area. Then a plan prefix is added to the mission and sent to the PA_i as "*plan*(*coverage* (*A*1))" for planning.

After the PA_i receives the message "*plan*(*coverage* (*A*1))", it decomposes the message into parts. Using the assigned network to *A*1, it plans coverage path for all robots in the team. During the planning, it firstly checks for the required arc set. If all arcs are in the required arc set, the CPP solution technique is used to construct the tour. If all arcs are not in the required arc set, then the RPP solution technique is used to construct the tour. Finally, this tour is partitioned among the robots considering their energy capacities using the MUA. Then, the PA_i decomposes its own plan from the generated plans and transforms the plan into a motion plan message as "*motionPlan* ($move(X_1, Y_1, Th_1)$, $move(X_2, Y_2, Th_2)$, ..., $move(X_n, Y_n, Th_n)$)" and sent to the AA_i for performing the plan. In this message, expression "$move(X_n, Y_n, Th_n)$" means that the robot should move to (X_n, Y_n) coordinate and the robot's heading should be Th_n after robot finished its move at the specified coordinate. Also, other planning agents (PA_j) have to be informed about generated plans. So that, PA_i prepares *allRobotPlans*(*plan*(ID_1, $robotPlan_1$, $arcsToBeCovered_1$), ..., *plan*(ID_n, $robotPlan_n$, $arcsToBeCovered_n$)) message and sent to the CA_j via the CA_i. This message contains all the robot plans generated by the MUA. Each robot plan contains three types of information which are unique robot id, sequence of vertices (coverage path plan), and whether an arc traversed by coverage or deadheading. After the CA_i receives *allRobotPlans* message from the PA_i, it sends to the CA_j (where, $j = 1, \dots, n, j \neq i$) via the TCP/IP communication channel. Then, the CA_j transfers this message to the PA_j. Similar to the PA_i, the PA_j decomposes its own plan from the received message and turns it into a motion plan and sends to the AA_j. Experimental results of the proposed method are given in details in the following section.

4. Experimental results

The algorithms in the proposed approach were coded in C++ and tested using the Pioneer 3-DX robots both in the laboratory and in MobileSim simulation environments. The developed agent-based robot control architecture was used in applications both in laboratory environment and MobileSim simulations. The P3-DX robots have onboard P3-800 computer with Linux OS, sonar sensors, PTZ camera, wireless network card, and the SICK LMS Laser range finder. Mainly the SICK LMS Laser range sensor is used for the coverage task. The sensor has normally a range of 50 m. Nevertheless, in this application, due to much smaller dimensions of the environment, the range is restricted to 3 m with software. With this range, when the robot passes through a point, it covers every point inside a semicircle of 3 m radius.

As explained before, there are two different energy consumptions: traveling and coverage energy. During the experiments, traveling energy related with motor, embedded computer, microcontroller card, and using sonar for obstacle avoidance is modeled as $e_{ij} = (17.49+(7.4\cdot(\text{velocity}/1000)))\cdot t_{ij}$ where $t_{ij} = d_{ij}/\text{velocity}$ is travel time. The energy consumption of the SICK LMS-200 laser range finder, which is related to coverage, is $q_{ij} = 20 \cdot t_{ij}$. The coverage energy is calculated as the sum of the traveling energy and the energy consumed by the SICK LMS-200. Velocity of the robots is set to a constant value of 400 mm/s. Hence, each robot consumes 0.051 J/mm for traveling and 0.101 J/mm for coverage. The results of the experiments conducted at ESOGU Artificial Intelligence & Robotic laboratory [27] and the MobileSim simulation environment are given in the separate subsections.

4.1. An application in the laboratory environment

In order to test the proposed algorithm, indoor environment test bed is constructed in the laboratory as shown in Fig. 7. A topological map of this environment with a GVD-based network is also given in Fig. 8. Since all arcs are required to be covered, the CPP approach is used to construct a tour as follows: (**1**–**2**), (**2**–**6**), (**6**–**7**) (**7**–**8**), (**8**–**9**), (**9**–**10**), (**10**–**11**), (**11**–**8**), (**8**–**14**), (**14**–**15**), (**15**–**16**), (**16**–**13**), (**13**–**14**), (*14*, *13*), (**13**–**12**), (**12**–**17**), (**17**–**18**), (**18**–**19**), (**19**–**20**), (**20**–**17**), (*17*–*12*), (**12**–**7**), (*7*–*6*), (**6**–**5**), (**5**–**4**), (**4**–**3**), (**3**–**1**).

The robot performs coverage task while passing through the bold-written arcs and use the italic-written arcs for deadheading. If there were only one robot with proper energy capacity, the robot would cover the environment by consuming 4162.94 J. In this case, tour length of the robot is 42 931.5 mm (the optimal solution) and the coverage time of the environment is 282 s.

Next, two P3-DX robots with energy capacity of 3200 J at node 1 (charging point) are used to cover the environment. The tour

Fig. 7. A photograph of the test environment.

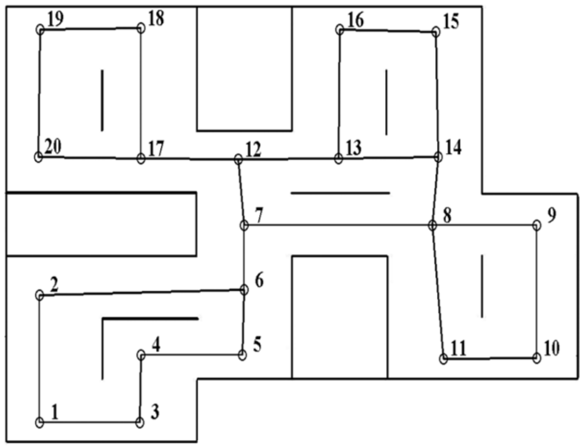


Fig. 8. Topological map and GVD-based network of the test environment.

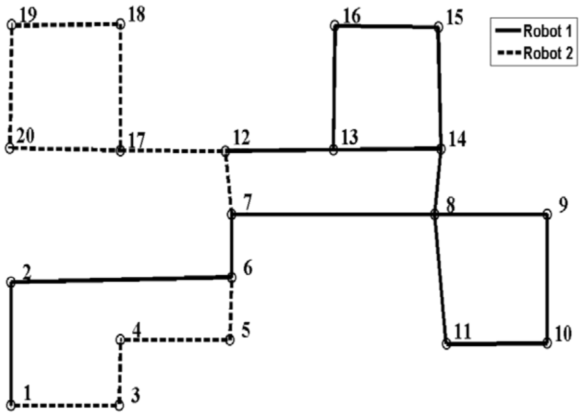


Fig. 9. Partitioning the tour among two robots.

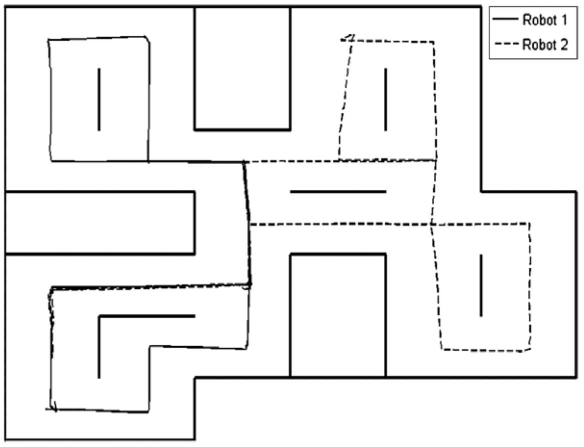


Fig. 10. Trace of the mobile robot (drawn using x–y coordinates from real data).

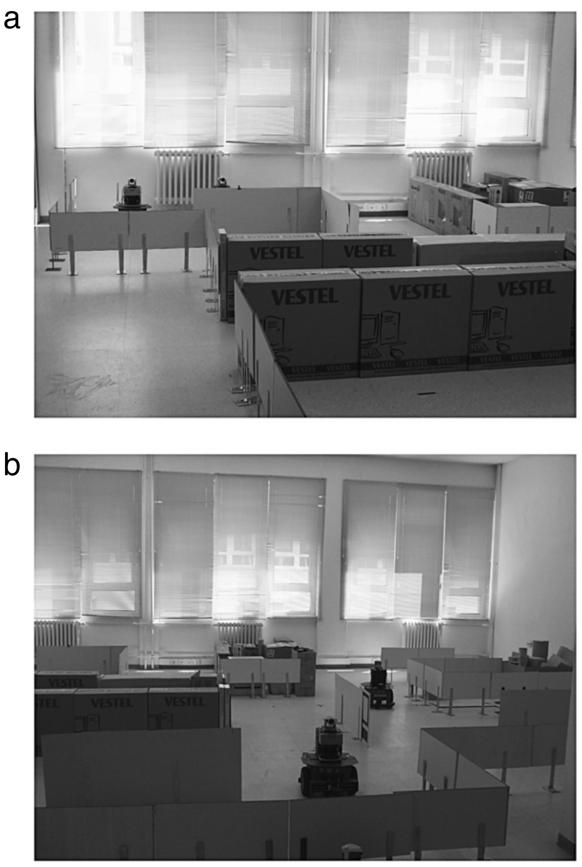


Fig. 11. Snapshots while P3-DX mobile robots cover the area. (a) $t = 0$ s, (b) $t = 60$ s.

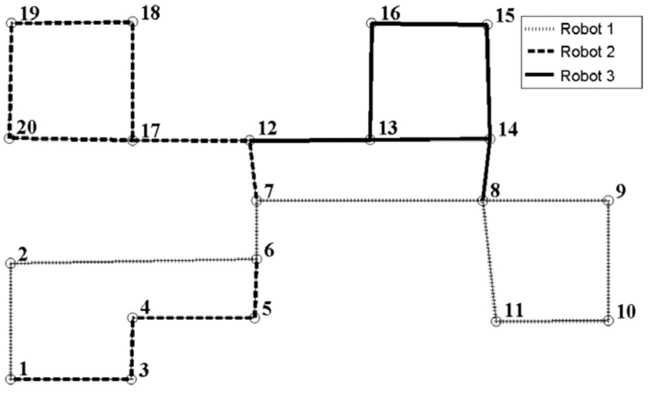


Fig. 12. Partition of the tour among three robots.

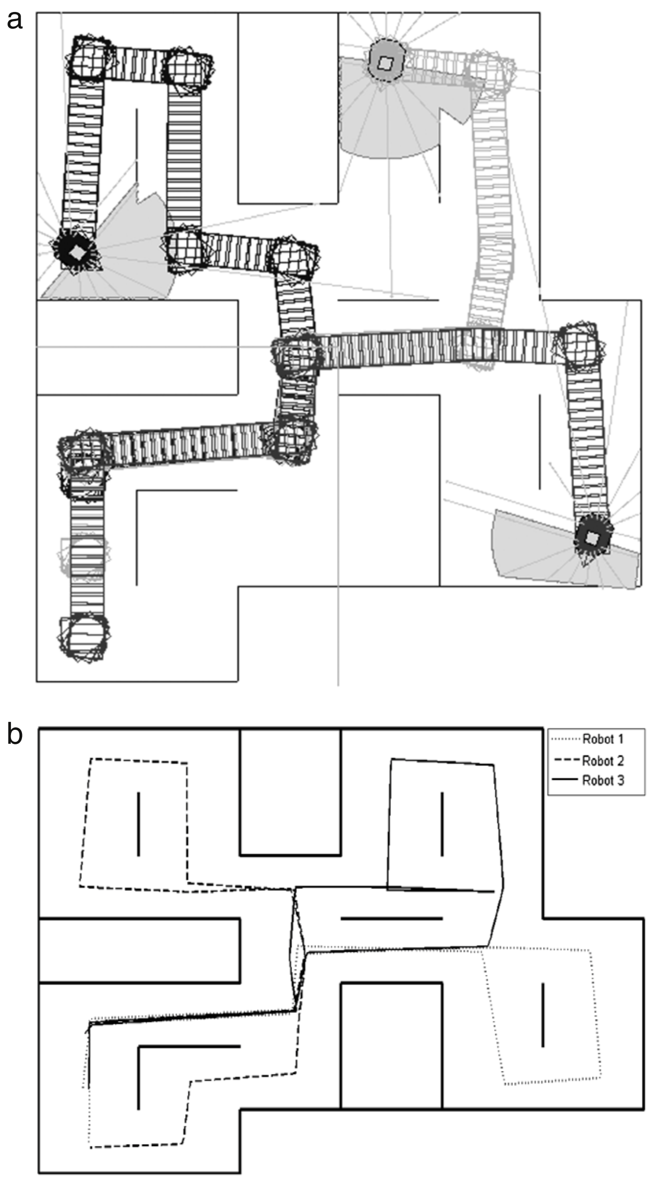


Fig. 13. (a) MobileSim screenshot and (b) Traces for three-robot coverage application.

given above is partitioned among the robots by using the proposed algorithm. The obtained tours are as follows.

The tour for robot 1: (**1–2**), (**2–6**), (**6–7**) (**7–8**), (**8–9**), (**9–10**), (**10–11**), (**11–8**), (**8–14**), (**14–15**), (**15–16**), (**16–13**), (**13–14**), (*14*, *13*), (**13–12**), (*12–7*), (*7–6*), (*6–2*), (*2–1*), and

The tour for robot 2: (*1–2*), (*2–6*), (*6–7*), (*7–12*), (**12–17**), (**17–18**), (**18–19**), (**19–20**), (**20–17**), (*17–12*), (**12–7**), (*7–6*), (**6–5**), (**5–4**), (**4–3**), (**3–1**).

The solution obtained by the proposed approach for two robots is shown in Fig. 9. Tour lengths are 33 352 mm and 23 946.9 mm and robots consume 2953.03 J and 1944.44 J, respectively. Totally, robots traverse 57 298.9 mm and consume 4897.47 J. Coverage time of the environment is 228 s. Note that the coverage time for multi-robot case is defined as the longest task completion time of any robot. Comparing to the single-robot case, the coverage time is decreased by 19% whereas the distance and the consumed energy increased by 33% and 17%, respectively.

Table 1
The results of the proposed planning for larger test environment.

#Robots	TTL (mm)	TCE (J)	ATL (mm)	ACE (J)	MTL (mm)	MCE (J)	Solution time (ms)
1	415 106.0	35 975.9	415 106.0	35 975.9	415 106.0	35 975.9	912.0
2	476 105.0	39 094.5	238 052.5	19 547.3	334 207.0	27 364.4	913.0
3	572 074.0	44 000.8	190 691.3	14 666.9	210 503.0	15 852.8	935.0
4	581 271.2	44 471.1	145 317.8	11 117.8	184 193.0	12 843.6	908.0
5	700 750.0	50 579.6	140 150.0	10 115.9	158 249.0	11 730.1	902.0
6	813 464.0	56 342.0	135 577.3	9 390.3	151 877.0	9 920.4	883.0
7	899 075.5	60 718.9	128 439.4	8 674.1	151 530.0	9 699.6	910.0
8	1021 670.1	66 986.5	127 708.8	8 373.3	153 215.0	9 087.0	892.0
9	1112 978.5	71 654.6	123 664.3	7 961.6	141 433.0	8 433.5	910.0

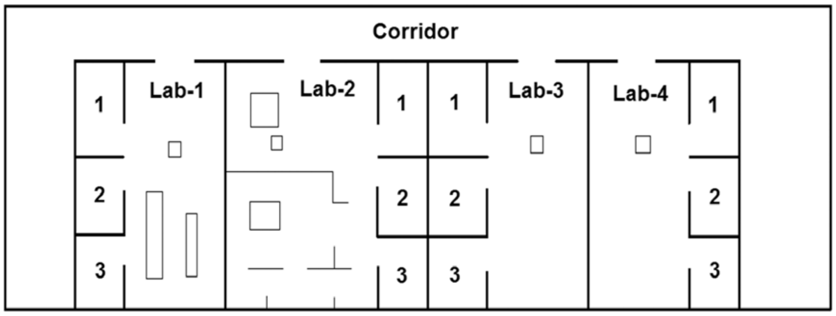


Fig. 14. Topological map of the first floor.

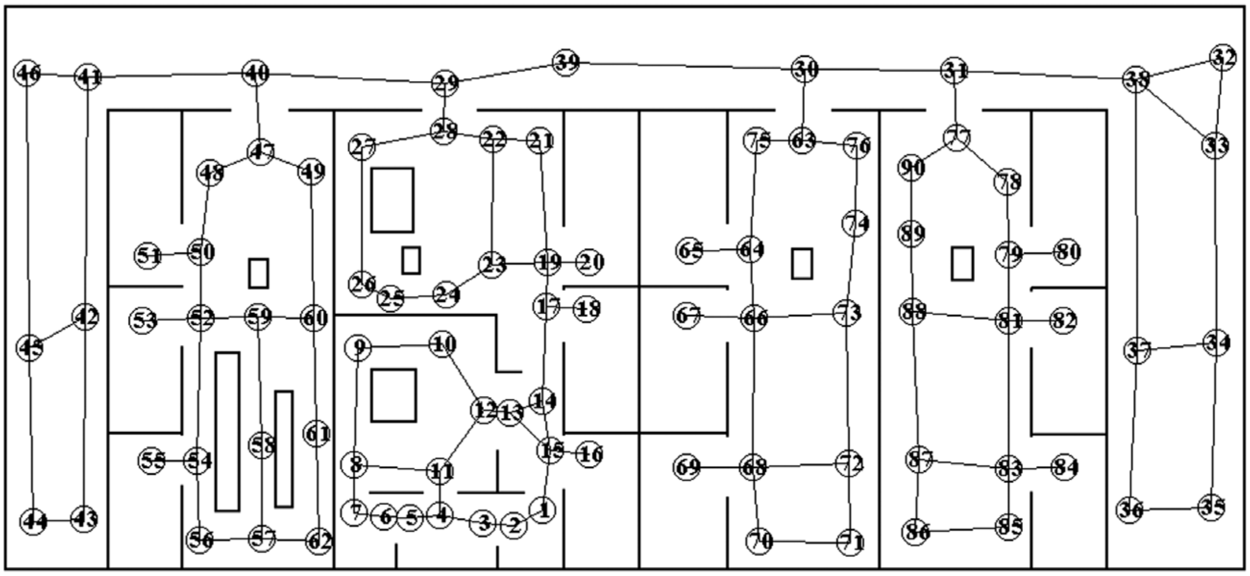


Fig. 15. GVD-based network of the first floor.

Fig. 10 shows the traces of the robots' motion during the application in the laboratory. Fig. 11 shows two snapshots while the P3-DX mobile robots cover the area. A video of the live performance of the mobile robots is recorded and can be reached from the following web site: www.ai-robotlab.ogu.edu.tr.

In another application, three robots are used to test the proposed algorithm in the MobileSim simulation environment. The same software architecture and robot types are used in the experiments. The partitioned tours for three robots of energy capacities of 2000 J are:

The tour for robot 1: (**1–2**), (**2–6**), (**6–7**), (**7–8**), (**8–9**), (**9–10**), (**10–11**), (**11–8**), (*8–7*), (*7–6*), (*6–2*) and (*2–1*) (Dotted Line).

The tour for robot 2: (*1–2*), (*2–6*), (*6–7*), (*7–12*), (**12–17**), (**17–18**), (**18–19**), (**19–20**), (**20–17**), (*17–12*), (**12–7**), (*7–6*), (**6–5**), (**5–4**), (**4–3**), and (**3–1**) (Dashed Line).

The tour for robot 3: (*1–2*), (*2–6*), (*6–7*), (*7–8*), (**8–14**), (**14–15**), (**15–16**), (**16–13**), (**13–14**), (*14–13*), (**13–12**), (*12–7*), (*7–6*), (*6–2*) and (*2–1*) (Solid Line).

Fig. 12 shows the result of the proposed algorithm. Robots consume 1990.33, 1944.44 and 1807.44 J, respectively. Length of each tour is 23 466.8 mm, 23 946.9 mm, and 26 108.4 mm, respectively. Total consumed energy and total traversed distance are 5742.21 J and 73 520.8 mm, respectively. Coverage time of the environment is 168 s. Comparing to the single-robot case; the coverage time decreased by 40%, but the distance and the consumed energy increased by 71% and 37%, respectively.

Fig. 13 shows the traces and a screenshot of the mobile robots during the application in the MobileSim simulation environment.

Additionally, compared to two-robot case, the total consumed energy increased approximately 18% while maximum traveled distance decreased by 22%. A detailed analysis of these variations

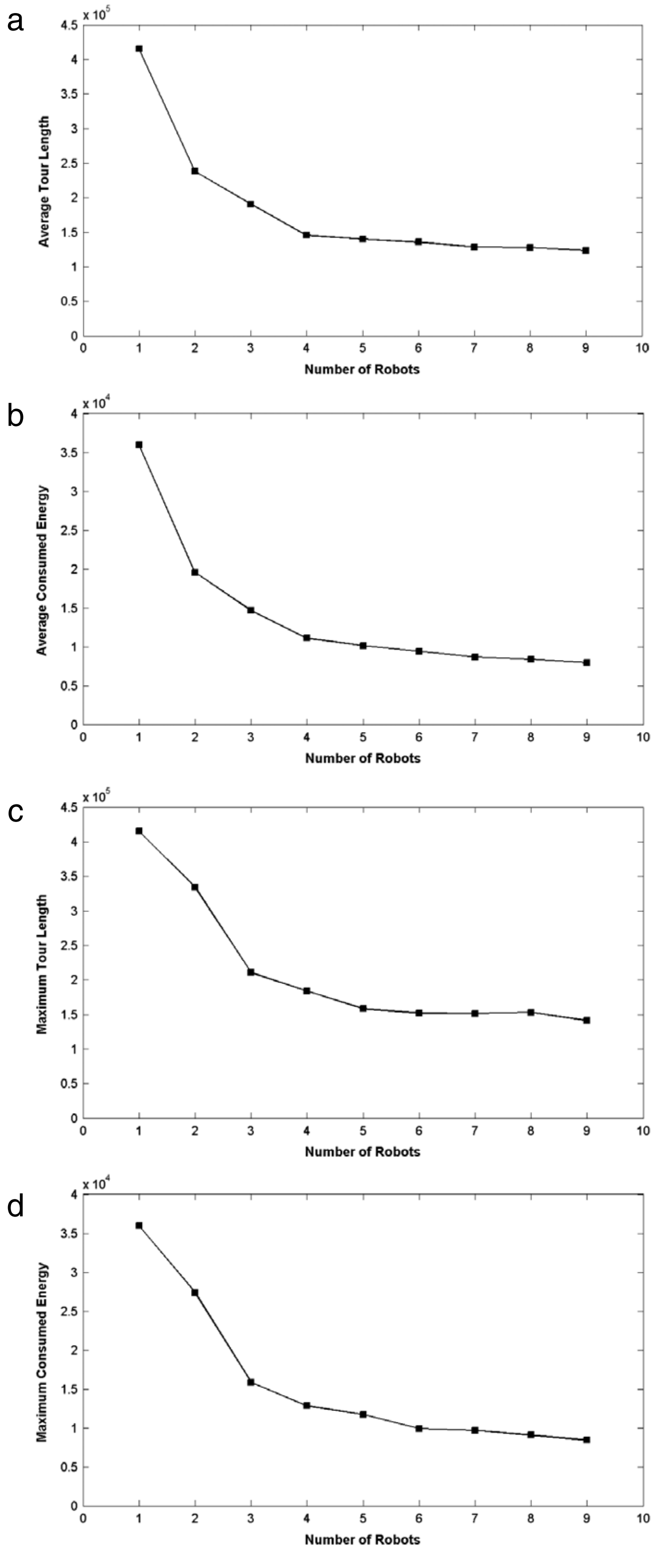


Fig. 16. The average tour length (a), average consumed energy per robot (b), maximum tour length (c), and maximum consumed energy (d) versus number of robots.

for more robots and a larger test environment is given in the following subsection.

4.2. *Test of the algorithm on a larger graph*

In order to test the proposed algorithm on a large-number-vertices environment, the first floor of the Eskisehir Osmangazi University Electrical Engineering laboratory building is used as the test bed. In this floor, there are 4 laboratories. Inside each laboratory, there are three rooms, tables, storage cabinets, and columns. A corridor connects the laboratories. Topological map of the first floor is given in Fig. 14. Number of vertices in the graph is 90 and depot vertex number is "44".

The corresponding network given in Fig. 15 is transferred to the MobileSim simulation environment. Simulations are carried out related to the planning up to nine robots using the proposed approach. Table 1 shows the results of the proposed planning for the larger test environment. In this table, Total Tour Length (TTL), Total Consumed Energy (TCE), Average Tour Length (ATL), Average Consumed Energy (ACE), Maximum Tour Length (MTL), Maximum Consumed Energy (MCE) and solution time vs. number of robots are given. Note that the solution is obtained in almost 0.9 s for each case.

As seen from the table, TTL and TCE increase linearly as the number of the robots increases. On the other hand, ATL, ACE, MTL, and MCE decrease. To see the effects of the number of robots on ATL, ACE, MTL, and MCE, diagrams in Fig. 16 are plotted.

As seen from Fig. 16, marginal utility of additional robot decreases in the case of average tour length per robot after 4 robots. Similarly, after 5 robots, marginal utility of adding a new robot decreases in all other cases. Therefore, maximum five robots may be assigned to the above coverage task. This kind of analysis can be used to determine the maximum number of robots to assign for a given coverage task.

5. Conclusion

In this study, a new approach based on the CARP was proposed and applied to MRSBCP for narrow interior environments. The proposed algorithm was developed by modifying the Ulusoy partitioning algorithm. To implement the proposed algorithm, agent-based robot control architecture was developed. Experiments in laboratory and simulation for small and large environments were conducted.

Although the solution of the proposed approach is not optimal, the solution was obtained in less than 1 s for a problem having 9 robots and 90 vertices. This is very important, since obtaining suitable tours for multi-robot teams in real-time applications in a short time is crucial. This approach can also be used to determine the maximum number of robots to assign for a given coverage task for efficient coverage which is very important for resource allocation.

Another important contribution of this study is that both the proposed algorithm and agent-based robot control architecture are flexible and can be used in other robot application problems. For instance, the study could be extended to consider time constrained sensor-based mobile robot coverage problem.

Acknowledgement

This work is supported by the Scientific and Technical Research Council of Turkey (TUBITAK), Contract No. 107E064, project title: "Mobile Robot Route Planning for Complete Coverage of Dynamic Indoor Environments".

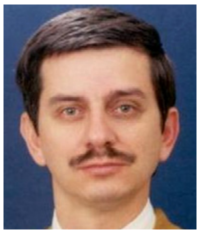
References

[1] H. Choset, Coverage for robotics—A survey of recent results, Annals of Mathematics and Artificial Intelligence 31 (2001) 113–126.
[2] E. Prassler, A. Ritter, C. Schaeffer, P. Fiorini, A short history of cleaning robots, Autonomous Robots 9 (2000) 211–226.
[3] E.M. Arkin, S.P. Fekete, J.S.B. Mitchell, Approximation algorithms for lawn mowing and milling, Computational Geometry 17 (2000) 25–50.
[4] C. Trevai, J. Ota, T. Arai, Multiple mobile robot surveillance in unknown environments, Advanced Robotics 21 (2007) 729–749.
[5] E.U. Acar, H. Choset, J.Y. Lee, Sensor-based coverage with extended range detectors, IEEE Transactions on Robotics 22 (1) (2006) 189–198.

[6] H. Choset, J. Burdick, Sensor-based motion planning: Incremental construction of the hierarchical generalized Voronoi graph, International Journal of Robotics Research 19 (2) (2000) 126–148.
[7] A. Sipahioğlu, A. Yazıcı, G. Kirlik, O. Parlaktuna, A sensor-based coverage approach in dynamic environments, in: Proceedings of 6th International Symposium on Intelligent and Manufacturing Systems, October 14–16, 2008, pp. 502–509.
[8] D. Kurabayashi, et al. Cooperative sweeping by multiple mobile robots, in: Proceedings IEEE Int. Conf Robotics Automat, vol. 3, 1994, pp. 1744–1749.
[9] D. Latimer, et al. Towards sensor based coverage with robot teams, in: Robotics and Automation Proceedings ICRA'02, vol. 1, 2002, pp. 961–967.
[10] Y. Mei, L. Yung-Hsiang, H.Y. Charlie, L.C.S. George, Deployment of mobile robots with energy and timing constraints, IEEE Transactions on Robotics and Automation 22 (3) (2006) 507–522.
[11] O. Parlaktuna, A. Sipahioglu, G. Kirlik, A. Yazici, Multi-robot sensor-based coverage path planning using capacitated arc routing approach, in: Proceedings of 24th IEEE International Symposium on Intelligent Control, ISIC09, Saint Petersburg, Russia, July 8–10, 2009, pp. 1146–1151.
[12] M. Dror (Ed.), Arc Routing Theory, Solution and Application, Kluwer Academic Publisher, Dordrecht, 2000.
[13] K.H. Rosen, Handbook of Graph Theory, CRC Press, New Jersey, 2004.
[14] J. Edmonds, E.L. Johnson, Matching, Euler tours and Chinese postman, Mathematical Programming 5 (1973) 88–124.
[15] J.K. Lenstra, A.H.G. Rinnooy Kan, On general routing problems, Networks 6 (3) (1976) 273–280.
[16] A. Hertz, G. Laporte, P.N. Hugo, Improvement procedures for the undirected rural postman problem, INFORMS Journal on Computing 11 (1999) 53–62.
[17] B.L. Golden, R.T. Wong, Capacitated arc routing problems, Networks 11 (1981) 305–331.
[18] A. Amberg, S. Voß, A hierarchical relaxations lower bound for the capacitated arc routing problem, in: Proceedings of the 35th Hawaii International Conference on System Sciences, vol. 3, 2002, p. 83b.
[19] J.M. Belenguer, E. Benavent, The capacitated arc routing problem: Valid inequalities and facets, Computational Optimization and Applications 10 (2) (1998) 165–187.
[20] J.M. Belenguer, E. Benavent, A cutting plane algorithm for the capacitated arc routing problem, Computers & Operations Research 30 (2003) 705–728.
[21] S. Wøhlk, New lower bound for the capacitated arc routing problem, Computers & Operations Research 33 (12) (2006) 3458–3472.
[22] A. Hertz, G. Laporte, M. Mittaz, A tabu search heuristic for the capacitated arc routing problem, Operations Research 48 (2000) 129–135.
[23] G. Ulusoy, The fleet size and mix problem for capacitated arc routing, European Journal of Operation Research 22 (1985) 329–337.
[24] D. Martin, A.J. Cheyer, D.B. Moran, The open agent architecture: A framework for building distributed software systems, Applied Artificial Intelligence 13 (1999) 91–128.
[25] R.R. Murphy, Introduction to AI Robotics, MIT Press, 2000, 0-262-13383-0.
[26] ARNL installation and operations manual, version 2.1, 2007.
[27] Eskiehir Osmangazi University, Artificial Intelligence and Robotic Laboratory, www.ai-robotlab.ogu.edu.tr.

Aydin Sipahioglu is an Assistant Professor in Industrial Engineering Department, Osmangazi University. He obtained his B.S. degree from Industrial Engineering Department, Istanbul Technical University, Turkey, in 1987. He obtained his Ph.D. degree in Operations Research at Osmangazi University, in 1996. He has been working at the Industrial Engineering Department of the Eskisehir Osmangazi University since 1993. His research interests are Integer Programming (especially TSP and CPP type routing problems), Logistics Management and Project Management.

Gokhan Kirlik obtained the B.S. degree from the Department of Industrial Engineering at Eskisehir Osmangazi University, Turkey, 2007. He also graduated from the Department of Computer Engineering as a second major program at Eskisehir Osmangazi University, Turkey, 2008. He is studying under the Master's of Operations Research Program at Eskisehir Osmangazi University. Also, he has been working at Eskisehir Osmangazi University Artificial Intelligence and Robotic laboratory since September 2007. His research interests are routing problems, multi-agent programming, and mobile robots.



Osman Parlaktuna is an Associate Professor of Electrical–Electronics Engineering at Eskisehir Osmangazi University, Turkey. He earned B.Sc. and M.Sc. Degrees in Electrical Engineering from Middle East Technical University, Ankara Turkey, and the Ph.D. from Vanderbilt University, Nashville, TN. He is the founder of Eskisehir Osmangazi University Artificial Intelligence and Robotic laboratory (http://www.ai-robotlab.ogu.edu.tr/). His research interests include mobile robotics, automation and control.

Ahmet Yazici obtained the B.S. degree from Osmangazi University, Turkey, 1998. He obtained M.S. and Ph.D. degrees in the area of Control Systems in 2000 and 2005. He has been working at the Computer Engineering Department of the Eskisehir Osmangazi University since 2005. He is also a co-founder of Eskisehir Osmangazi University Artificial Intelligence and Robotic laboratory (http://www.ai-robotlab.ogu.edu.tr/). His research interests are Control theory, Artificial Intelligence, and Mobile Robots.