Feedback coordination of limited capability mobile robots

Praneel Chand*

School of Engineering and Computer Science,
Victoria University of Wellington (VUW),
Wellington, New Zealand
and
School of Engineering and Physics,
The University of the South Pacific,
Suva, Fiji
E-mail: Praneel.Chand@usp.ac.fj
*Corresponding author

Dale A. Carnegie

School of Engineering and Computer Science,
Victoria University of Wellington (VUW),
Wellington, New Zealand
E-mail: Dale.Carnegie@vuw.ac.nz

Abstract: In some multi-robot applications, such as exploration, predefined task allocation and coordination can fail to function adequately. This failure is attributed to the inability to completely model a robot's interactions with the environment before task execution. Hence, the main focus of this paper is on a feedback coordination mechanism that executes periodically after initial task allocation. This feedback mechanism monitors the individual and group performance of worker robots. If the performance of a worker robot is unsatisfactory, a task reallocation algorithm adjusts the task–robot combinations of the team. Three cases of unsatisfactory robot performance that can be detected by the feedback mechanism include: complete failure, partial failure and poor performance. The task allocation and coordination strategy are applied to a multi-robot exploration task. Initial results from experiments in various types of environments indicate that the feedback coordination mechanism successfully identifies the three forms of robot failure, and improves the system's performance.

Keywords: multi-robot; limited capability; task allocation; coordination; exploration; map-building.

Reference to this paper should be made as follows: Chand, P. and Carnegie, D.A. (2010) 'Feedback coordination of limited capability mobile robots', *Int. J. Intelligent Systems Technologies and Applications*, Vol. 8, Nos. 1-4, pp.144–157.

Biographical notes: Praneel Chand is an Assistant Lecturer in the School of Engineering and Physics at the University of the South Pacific (USP), Suva, Fiji. He is also a part-time Doctoral Candidate at Victoria University of Wellington, Wellington, New Zealand. He completed PGD EngTech and B.Tech in Electrical and Electronics Engineering at USP. He is a Student

Member of the IEEE and its Robotics and Automation Society (RAS). His research interests include co-operative mobile robot systems, mobile robot navigation and autonomous control systems.

Dale A. Carnegie received his MSc with first class honours in Physics and Electronics and his PhD in Computer Science from the University of Waikato in Hamilton, New Zealand. Currently, he is a Professor of Computer Systems Engineering at Victoria University of Wellington, New Zealand where he heads the Mechatronics Research Group and co-ordinates the Computer Systems Engineering Programme. His research interests are in the areas of autonomous mobile robotics, sensors, embedded controllers and applied artificial intelligence.

1 Introduction

There are numerous applications for multi-robot systems such as search and rescue, exploration, and object manipulation. Benefits such as robustness to failure and increased efficiency can be achieved in multi-robot systems by distributing the group task across the team. Many multi-robot systems employ task allocation and coordination mechanisms to achieve these benefits. Task allocation mechanisms distribute tasks between different robots (Gerkey and Mataric, 2004). Coordination mechanisms allow individual robots within a group to take each others' actions into consideration such that the team operates coherently (Farinelli et al., 2004). Coalition formation (Parker and Tang, 2006; Vig and Adams, 2006) employs task-allocation methods that allow multiple robots to collectively achieve the objectives of a task that the individual robots are incapable of executing.

A hierarchical heterogeneous multi-robot system for urban search and rescue (USAR) is currently under development (Williamson and Carnegie, 2006). This multi-robot system has three categories of robots: grandmothers, mothers and daughters. At the top of the hierarchy, the grandmothers are physically the largest and most powerful computationally. Grandmother robots are generally employed to manage the operation of a group task. The lower-tiered robots (mothers and daughters) are smaller in size and less computationally powerful. Mothers and daughters are also limited in their sensing and acting abilities. The smaller size of the mother and daughter robots enables them to be deployed for searching the environment as worker robots.

Allocating tasks to robots in a heterogeneous multi-robot system such as Williamson and Carnegie (2006) requires a strategy that takes into account the capabilities of the different robots. A task-allocation method that employs numerical robot capability data to determine robot suitability for a task is presented in Chand and Carnegie (2007a). Four major categories of robot capabilities (i.e. resources) represented in the approach include processing, communication, sensing and actuation.

However, after initial task allocation the robots may not perform as desired due to the inability to fully model all interactions with the environment accurately. Hence, the focus of this paper is on a feedback coordination mechanism that can be employed to monitor the efficiency of robot resource use during task execution. By appropriately responding to robot failures such a mechanism could allow a group task to be carried out with increased efficiency.

2 Related work

A number of task allocation and coordination strategies for multi-robot systems have been reviewed in Gerkey and Mataric (2004) and Farinelli et al (2004). While Gerkey and Mataric (2004) review task-allocation methods, Farinelli et al (2004) classifies multi-robot systems based on coordination. A review of relevant task allocation, coordination, team building and coalition formation techniques is presented in Chand and Carnegie (2007a).

Fault tolerance in multi-robot systems can be viewed as a specialised form of multi-robot coordination. In fault tolerant multi-robot systems, performance metrics and monitors are used to detect robot failures. The team responds to individual robot failures by dynamically reselecting or reassigning the tasks of failed robots.

Parker (1997) implemented performance monitors for behaviour sets in the L-ALLIANCE architecture. Task completion time is used as the performance metric in this architecture. Task reallocation (TR) is effected via a learning process that updates the control parameters of the behaviour sets associated with the task that a robot executes.

Kannan and Parker (2006) developed metrics to investigate the influence of fault tolerance on overall system performance. In their implementation, the robots are required to perform a number of tasks and each robot–task pair contributes towards the overall performance. Experimental data quantify the performance of a complex heterogeneous multi-robot application.

An extension to Kannan and Parker's (2007) work on fault tolerance measures the effectiveness of fault tolerance in box pushing and deployment tasks. Fault tolerance in these tasks is tested using predefined and adaptive causal model methods. The implementation of causal model methods can be cumbersome when there are many robots, fault nodes and fault combinations. Additionally, causal model methods need to be tailored for the task that the robots execute and environment that they operate in.

Tolerance to sensor failures in a small team of distributed robots has been investigated in Long et al. (2003). This research extends the sensing fault tolerance capability of the Sensor Fusion Effects (SFX-EH) architecture (Murphy and Hershberger, 1999) to multiple robots. Sensor failures are diagnosed by allowing the robots to share knowledge of state of their sensors and task execution via communication. Tasks are redistributed when robots become inoperable. The main drawback of this implementation is that it only addresses sensing failures. Also, the implementation uses cameras and it would be difficult to apply this technique to robots without visual sensing.

The work presented in this paper varies significantly from those reported in the literature. We require a feedback mechanism that monitors the processing, communication, sensing and actuation resources used by a robot towards a task. An individual robot's resource utilisation needs to be adjusted using both local and global performance information. Resource utilisation is represented by the control algorithm execution rate (or robot processor usage) towards each physical resource type for a particular task.

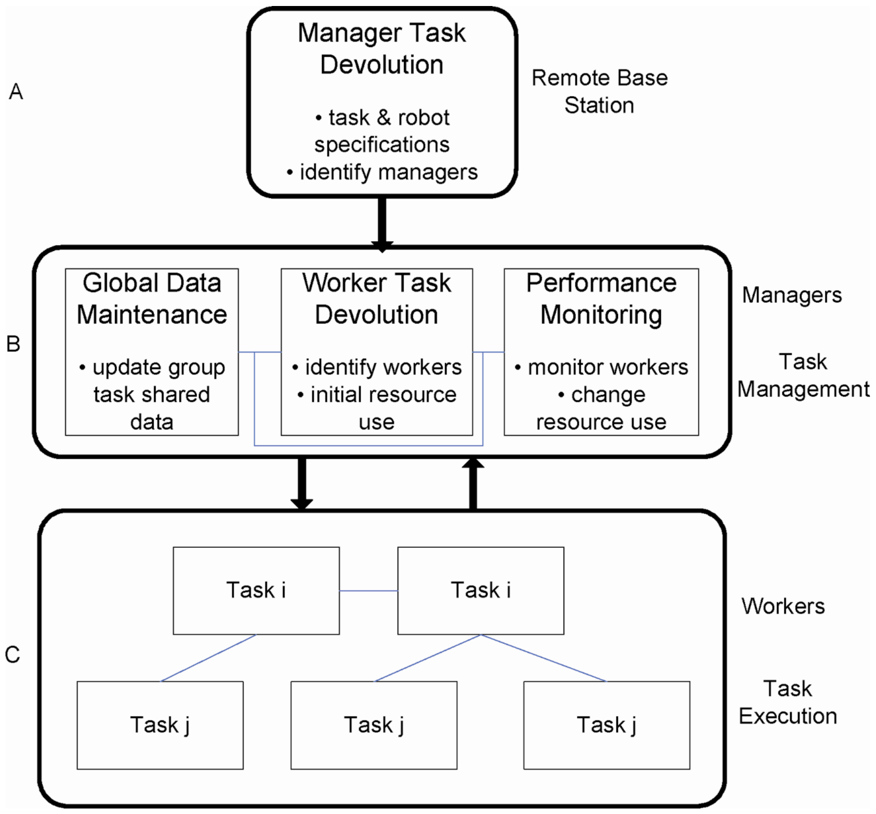
In this paper, TR is considered equivalent to adjusting a robot's resource utilisation. However, resource utilisation adjustment can also optimise the current task that a robot executes.

3 System overview

A block diagram representation of the task allocation and coordination mechanism employed in our multi-robot system is shown in Figure 1. There are three levels of control. At the highest level (level A), a remote base station computer is used to specify the group task and robots. Depending on the management task requirements and robot capabilities, the remote computer identifies manager robot(s) (level B). The manager robots are delegated the responsibilities of global data maintenance, worker task devolution and performance monitoring. After manager task devolution, the remote computer is no longer required by the robot team since task management is essentially transferred to the manager robot(s).

At the third level of control (level C), worker robots are responsible for executing the objectives of the group task. Worker robots are selected by the manager robot(s) during execution of the worker task devolution process (Section 4). Depending on the nature of worker tasks to be allocated, the worker robots are assigned a position in a predefined hierarchy. Worker robots executing some tasks (e.g. task *i*) could be supervising robots executing other tasks (e.g. task *j*).

Figure 1 Overview of the multi-robot task allocation and coordination mechanism (see online version for colours)



Following the worker task devolution process, the worker robots perform their tasks while the manager robots monitor their performance. To monitor performance, the manager robot(s) determine the efficiency of the worker robots resources during task execution. If the performance of an individual robot (hence team) is not satisfactory, the worker robots' resource utilisation is adjusted by a feedback coordination mechanism (Section 6). Resource utilisation adjustment enables TR and can cause a worker robot's position in the hierarchy to change. An adapted version of the worker task devolution process is employed for TR.

The feedback coordination mechanism is demonstrated on a multi-robot exploration task (Section 5) with results and conclusions presented in Sections 7 and 8, respectively.

4 Worker task devolution

The worker tasks are manually specified by a human user via a remote base station computer and transferred to the manager robot(s) during initial task devolution. Numerical vectors of task requirements (VOTR) (Chand and Carnegie, 2007a) represent the resource requirements for the worker tasks. Robot capabilities are expressed by numerical vectors of merit (VOM) that represent processing, communication, sensing and actuation resources available on a robot.

The capability of resource type $\mathrm{RC}_{\mathrm{type}}$ for each task t_i and robot r_j is specified as:

$$\mathrm{RC}_{\mathrm{type}} = \left\{ \mathrm{rc}_{\mathrm{type}1}, \mathrm{rc}_{\mathrm{type}2}, \ldots, \mathrm{rc}_{\mathrm{type}k}, \ldots, \mathrm{rc}_{\mathrm{type}q} \right\} \tag{1}$$

where $\mathrm{type} \in \{\mathrm{proc, comm, sense, act}\}$; $\mathrm{rc}_{\mathrm{type}\,k}$ is the kth sub-resource type and q is the number of sub-resources for resource $\mathrm{RC}_{\mathrm{type}}$.

For each resource type, the sub-resources are:

- *Processing*: benchmark and memory.
- *Communication*: bandwidth and range.
- *Sensing*: a score representing each type of sensor present on the robot.
- *Actuation*: battery capacity, base size, base performance and manipulator.

For each task, a set of weights $\mathrm{KC}_{\mathrm{type}}$, comprising sub-resource weights $\mathrm{kc}_{\mathrm{type}k}$, is also specified. These weights are utilised with $\mathrm{RC}_{\mathrm{type}}$ to bias the sub-resources $\mathrm{rc}_{\mathrm{type}k}$ during task allocation.

$$\mathrm{KC}_{\mathrm{type}} = \left\{ \mathrm{kc}_{\mathrm{type}1}, \mathrm{kc}_{\mathrm{type}2}, \ldots, \mathrm{kc}_{\mathrm{type}k}, \ldots, \mathrm{kc}_{\mathrm{type}q} \right\} \tag{2}$$

Initial worker robot selection is a greedy assignment process that utilises a vector of task suitability (VOTS) representing each robot's suitability for a particular task.

The *j*th robot's suitability for the *i*th task is expressed as VOTS_{ij}. Based on the sub-resources required for the task $t_i\text{rc}_{\text{type}k}$ and sub-resources available on robot $r_j\text{rc}_{\text{type }k}$, the VOTS data for each resource type $\text{VOTS}_{\text{type}ij}$ can be computed by (3). Task t_i can be executed by robot r_j if $r_j\text{rc}_{\text{type }k} \geq t_i\text{rc}_{\text{type }k}$ for all resource and sub-resource types.

$$\text{VOTS}_{\text{type }ij} = \sum \text{kc}_{\text{type }k} \frac{r_i\text{rc}_{\text{type }k}}{t_i\text{rc}_{\text{type }k}} \tag{3}$$

The procedure for assigning tasks to worker robots after obtaining VOTS data are outlined below:

1 Identify a subset of robots that are able to execute each worker robot task.

2 Determine a weighted sum of the $\text{VOTS}_{\text{type }ij}$ data for each capable task–robot combination. This weighted sum represents a robot's load handling capacity LH for that task.

3 Count the number of other tasks that each capable robot can execute. This value represents the robot's task diversity capability TD.

4 Determine each robot's value *V* of performing each task using the LH and TD values from steps 2 and 3, respectively. The balance between the LH and TD variables is controlled by weights w_{LH} and w_{TD}.

5 For each task, sort the capable robots in descending order of value *V* data.

6 Based on the number of capable robots for each task, sort the tasks in ascending order.

7 Store the sorted capable robots and tasks in a robot–task capability matrix for TR use.

8 Using the worker task requirements, select and assign the quantity of robots needed for each type of task.

9 For each robot that is allocated a task, initialise the corresponding resource utilisation values.

$$V = w_{\text{LH}} \times \text{LH} + w_{\text{TD}} \times \text{TD} \tag{4}$$

For each task–robot combination, the resource utilisation of each resource category $\text{RU}_{ij\text{cat}}$ is expressed as:

$$\text{RU}_{ij\text{ cat}} = \left\{\text{ru}_{ij\text{ cat}1}, \text{ru}_{ij\text{ cat}2}, \ldots, \text{ru}_{ij\text{ cat}s}, \ldots, \text{ru}_{ij\text{ cat}z}\right\} \tag{5}$$

where $\text{cat} \in \{\text{plan}, \text{comm}, \text{sense}, \text{act}\}$; $\text{ru}_{ij\text{ cat}s}$ is the *s*th task-dependent resource utilisation sub-category and *z* is the number of sub-categories for resource utilisation $\text{RU}_{ij\text{ cat}}$.

5 Group task description

The team of robots is required to perform a multi-robot exploration task. Due to the limited processing and sensing capabilities of the worker robots, the exploration task is divided into planner and explorer tasks. Worker robots can be assigned one or both of these tasks. Depending on the worker robots processing and sensing abilities, the global world is subdivided into smaller local worlds.

Explorers are assigned local worlds to explore by the planner robots. A greedy algorithm trades off the utility and cost associated with reaching unexplored local worlds. The trade off is influenced by a worker explorer task score TS_{ej} (15). This parameter enables weaker explorers (slower robots) to favour local worlds close to their current location by multiplying neighbour local-world utilities by a utility factor uf_j (6). The feedback coordination performance data are also employed to estimate the completion time ECT_k of the *k*th explorer when there are less unexplored local worlds than explorer robots. This enables the best explorers to be selected for the remaining local worlds.

$$\mathrm{uf}_j = \begin{cases} \dfrac{1}{\mathrm{TS}_{ej}} & \text{; for explorer } j\text{'s neighbour areas} \\ \mathrm{TS}_{ej} & \text{; for other explorer's neighbour areas} \end{cases} \tag{6}$$

To move within and between local worlds, the individual robots use a navigation system that employs both reactive and deliberative control (Lee-Johnson et al., 2007). A two-tiered *A** path planning algorithm is employed by the planners to guide the explorers to their assigned local worlds (Chand and Carnegie, 2007b). The explorers use a line scanning technique to map out obstacles.

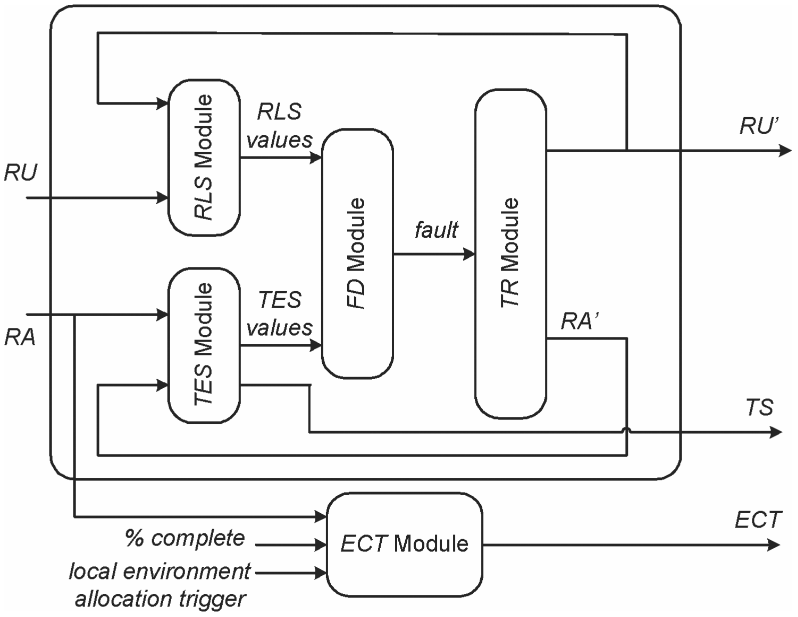
6 Feedback coordination mechanism

Feedback coordination is performed by the manager robots and executes periodically with time interval T_m after initial task allocation. If the feedback mechanism detects that the performance of a worker robot is not satisfactory, a task-reallocation process adapted from the worker task devolution is called. The TR algorithm applies a set of group task-specific rules that adjusts a robot's suitability for a particular task.

A block diagram of the feedback coordination mechanism is shown in Figure 2. The current resource achievement (RA) and corresponding current resource utilisation (RU) (or task allocation) for each worker robot are input to the feedback mechanism. Each robot's RA data are compared with expected (best-case scenario) RA′ data to compute the robot's task execution success (TES) value. The RU data of each robot are compared with expected RU′ data to compute the robot's load success (RLS) value. A failure detection (FD) module then takes the RLS and TES values as input. If a failure is detected, the TR module is triggered and the RU and achievement targets of the worker robots are updated.

It is assumed that the selected robots have sufficient processing resources for the multi-robot exploration task. Hence, the RLS values are not utilised in the feedback coordination. In the multi-robot exploration application, the feedback mechanism is also triggered to replace or remove explorer worker robots that become idle when no suitable unexplored areas remain.

Figure 2 Feedback coordination mechanism block diagram



6.1 *Performance monitoring*

The performance monitoring components of the feedback coordination mechanism include the TES, RLS and FD modules. All performance monitoring variables are normalised to unit interval values. The RU achievement of each resource category for each task–robot combination $\mathrm{RA}_{ij\,\mathrm{cat}}$ is specified as:

$$\mathrm{RA}_{ij\,\mathrm{cat}} = \{\mathrm{ra}_{ij\,\mathrm{cat}\,1}, \mathrm{ra}_{ij\,\mathrm{cat}\,2}, \ldots, \mathrm{ra}_{ij\,\mathrm{cat}\,s'}, \ldots, \mathrm{ra}_{ij\,\mathrm{cat}\,z'}\} \tag{7}$$

where $\mathrm{cat} \in \{\mathrm{plan}, \mathrm{comm}, \mathrm{sense}, \mathrm{act}\}$; $\mathrm{ra}_{ij\,\mathrm{cat}\,s'}$ is the s'th task-dependent RU achievement sub-category and z' is the number of sub-categories for RU achievement $\mathrm{RA}_{ij\,\mathrm{cat}}$.

Planning achievement is usually represented by the number of local and/or global plans made. Communication achievement can be represented by the volume of messages transmitted and received successfully. The accuracy of covered or explored area can represent sensing achievement. Actuation achievement may include criteria such as average speed, distance travelled or objects moved.

By integrating the instantaneous values of current and expected target achievements over N task monitoring time intervals, overall values of achievement $\mathrm{ORA}_{ij\,\mathrm{cat}}$ and expected achievement $\mathrm{ORA}'_{ij\,\mathrm{cat}}$ can be determined.

$$\mathrm{ORA}_{ij\,\mathrm{cat}} = \sum^{N} \mathrm{RA}_{ij\,\mathrm{cat}} \tag{8}$$

$$\mathrm{ORA}'_{ij\,\mathrm{cat}} = \sum^{N} \mathrm{RA}'_{ij\,\mathrm{cat}} \tag{9}$$

A set of weights $\mathrm{KA}_{i\,\mathrm{cat}}$, comprising achievement sub-category weights $\mathrm{ka}_{i\,\mathrm{cat}\,s'}$, is also specified for each task. These weights are used in conjunction with $\mathrm{RA}_{ij\mathrm{cat}}$ and $\mathrm{RA}'_{ij\,\mathrm{cat}}$ to bias sub-category achievements $\mathrm{ra}_{ij\mathrm{cat}s'}$ and $\mathrm{ra}'_{ij\,\mathrm{cat}\,\mathrm{s}}$ when determining success.

$$\mathrm{KA}_{i\,\mathrm{cat}} = \left\{\mathrm{ka}_{i\,\mathrm{cat}\,1}, \mathrm{ka}_{i\,\mathrm{cat}\,2}, \ldots, \mathrm{ka}_{i\,\mathrm{cat}\,s'}, \ldots, \mathrm{ka}_{i\,\mathrm{cat}\,z'}\right\} \tag{10}$$

To combine the success values of each resource category into a single TES value, a second set of bias weights $w_{i\mathrm{cat}}$ is used. The individual robot TES value $\mathrm{TES}_{ij\mathrm{ind}}$ is determined using only the individual robot's instantaneous achievement data (11). On the other hand, a robot's team TES value $\mathrm{TES}_{ij\mathrm{tm}}$ is determined as a fraction of the instantaneous achievements of all *n* robots in the team executing a task (12).

Replacing $\mathrm{RA}_{ij\,\mathrm{cat}}$ and $\mathrm{RA}'_{ij\mathrm{cat}}$ with $\mathrm{ORA}_{ij\mathrm{cat}}$ and $\mathrm{ORA}'_{ij\,\mathrm{cat}}$ respectively in (11), yields an overall individual robot success value $\mathrm{OTES}_{ij\,\mathrm{ind}}$. Making the same replacements in (12) produces a robot's overall team success value $\mathrm{OTES}_{ij\,\mathrm{tm}}$.

$$\mathrm{TES}_{ij\mathrm{ind}} = \sum w_{i\mathrm{cat}} \left(\sum ka_{i\mathrm{cat}s'} \frac{\mathrm{ra}_{ij\mathrm{cat}s'}}{\mathrm{ra}'_{ij\mathrm{cat}s'}} \right) \tag{11}$$

$$\mathrm{TES}_{ij\mathrm{tm}} = \sum w_{i\mathrm{cat}} \left(\sum ka_{i\mathrm{cat}s'} \frac{\mathrm{ra}_{ij\mathrm{cat}s'}}{\sum_{j=1}^{n} \mathrm{ra}'_{ij\mathrm{cat}s'}} \right) \tag{12}$$

Individual robot RLS values are determined in a similar manner to TES values. The major difference is that $\mathrm{RA}_{ij\mathrm{cat}}$ and $\mathrm{RA}'_{ij\,\mathrm{cat}}$ data are replaced with $\mathrm{RU}_{ij\mathrm{cat}}$ and $\mathrm{RU}'_{ij\,\mathrm{cat}}$ data, respectively. Weight values $\mathrm{KA}_{i\mathrm{cat}}$ and $w_{i\mathrm{cat}}$ can also be replaced with new data. RLS values verify that a robot has the capacity to process resource data in a timely manner.

The FD module detects faults when success values are below threshold. Three forms of robot failure where TR is necessary can be detected: complete failure, partial failure and poor performance. A complete failure CF_{ij} is detected when no achievement data are received from a robot and it has no pulse. Partial failures PF_{ij} occur when a robot fails at its current task due to faulty hardware but is not dead (i.e. achievement data are received from the robot). This situation can be detected when a robot's instantaneous task execution success value $\mathrm{TES}_{ij\,\mathrm{ind}}$ is below a threshold value close to zero $\mathrm{TES}_{\mathrm{T}}$ (13). When a robot's overall task execution success value $\mathrm{OTES}_{ij\,\mathrm{ind}}$ is below a second non-zero threshold value $\mathrm{OTES}_{\mathrm{T}}$ (14), it is identified as a poor performance robot PP_{ij}.

$$\mathrm{PF}_{ij} = \begin{cases} 1; & \text{if } \mathrm{TES}_{ij\mathrm{ind}} < \mathrm{TES}_{\mathrm{T}} \\ 0; & \text{otherwise} \end{cases} \tag{13}$$

$$\mathrm{PP}_{ij} = \begin{cases} 1; & \text{if } \mathrm{OTES}_{ij\mathrm{ind}} < \mathrm{OTES}_{\mathrm{T}} \\ 0; & \text{otherwise} \end{cases} \tag{14}$$

Worker explorer task score TS_{ej} is determined from the overall team success value OTES_{ejtm}, where *e* corresponds to the index of the explorer task (15)

$$\mathrm{TS}_{ej} = \mathrm{OTES}_{ejtm} \tag{15}$$

Exploration ECT data (16) are computed using exploration time t_{ek}, unexplored area in the current local environment $\overline{A}_{ck}$, unexplored area in the new local environment $\overline{A}_{n'k}$, total area explored A_0 and predicted exploration rate E_0. Explored area and exploration rate data are obtained from sensing achievements.

$$\text{ECT}_k = \begin{cases} t_{ek}\left(\overline{A}_{ck} + \overline{A}_{n'k}\right)/A_0 & ;\ \text{if } t_{ek} > T_m \\ \left(\overline{A}_{ck} + \overline{A}_{n'k}\right)/E_0 & ;\ \text{otherwise} \end{cases} \tag{16}$$

6.2 Task reallocation

During the TR process a robot's VOTS derived value in the robot–task capability matrix is updated with new overall task execution success $\text{OTES}_{ij\text{ind}}$ data. The robots are then re-ranked in descending order using this updated value. After re-ranking, a set of rules is applied to further adjust robot rankings and remove faulty robots. These rules are outlined below and listed in order of priority:

1 Remove completely failed robots in CF from all tasks in the robot–task capability matrix.

2 Delete the currently assigned task *i* from the robot–task capability matrix for partially failed robots in PF.

3 Shift partially failed robots in PF to the bottom row of the robot–task capability matrix for all other tasks.

4 Move poor performance robots in PP to the bottom row of the robot–task capability matrix for the currently assigned task *i*.

5 Promote robots with any task-specific capability requirements to the top of the robot–task capability matrix. An example is exploring an environment with varied terrain types. In some situations, a very specific actuation type may be needed to explore some local environments.

6 Adjust the quantity of tasks to be assigned based on the progress of the group task. For example, robots can be removed as exploration nears completion.

To complete the TR process, steps 6–9 of the worker task devolution procedure (Section 4) are executed to adjust the RU of the worker robots.

Partially failed and poor performance robots are given lower rankings to place them below non-faulty task capable robots. This ensures that non-faulty robots are given a chance to execute tasks. Depending on their new ranking and group task requirements, partially failed and poor performance robots may be assigned an alternative task, keep their current task (poor performance robots only) or be removed from the team.

7 Results

The multi-robot exploration task, task allocation and feedback coordination have been implemented using MATLAB®2007a. Global worlds with randomly positioned obstacles at 5% and 10% density have been generated. Half of these worlds also had rectangular sections of boggy terrain that covered approximately 5% of the exploration area. All global worlds tested were 38.4 m× 38.4 m in size. The system has been evaluated for single planner–single explorer and single planner–three explorer worker robot teams. Table 1 lists the tested environment and robot configurations. The rows represent test configurations while the columns indicate the configuration details. Table 2 shows the configuration code details for the columns of Table 1. The system's response to randomly generated partial and complete failures has also been evaluated.

Table 1 Environment and robot test configurations

	A	*B*	*C*	*D*	*E*	*F*	*G*	*H*
1	0	1	1	Yes	Yes	Yes	No	No
2	5	1	1	Yes	Yes	Yes	No	No
3	0	1	1	No	Yes	Yes	No	No
4	5	1	1	No	Yes	Yes	No	No
5	0	1	1	No	No	Yes	No	No
6	5	1	1	No	No	Yes	No	No
7	0	1	3	Yes	Yes	Yes	No	No
8	5	1	3	Yes	Yes	Yes	No	No
9	0	1	3	No	Yes	Yes	No	No
10	5	1	3	No	Yes	Yes	No	No
11	0	1	3	No	No	Yes	No	No
12	5	1	3	No	No	Yes	No	No
13	0	1	3	Yes	Yes	No	Yes	No
14	5	1	3	Yes	Yes	No	Yes	No
15	0	1	3	No	No	No	Yes	No
16	5	1	3	No	No	No	Yes	No
17	0	1	3	Yes	Yes	No	No	Yes
18	5	1	3	Yes	Yes	No	No	Yes
19	0	1	3	No	No	No	No	Yes
20	5	1	3	No	No	No	No	Yes

Table 2 Configuration code details

A	Bog density (%)
B	Number of planners
C	Number of explorers
D	TR feedback coordination enabled
E	Task score feedback enabled
F	Poor performance test
G	Partial failure test
H	Complete failure test

Figure 3 compares the exploration time and area explored for the various configurations tested. A simulation timeout of 5,000 sec (83.3 min) and 15,000 sec (250 min) has been used for the single planner–single explorer and single planner–three explorer tests, respectively. Poor performance, partial failure and complete failure experiments in all test environments indicate that the feedback coordination TR reduces exploration time and maximises explored area.

In the single planner–single explorer tests (1–6), the full feedback experiments (1–2) take on average 19.5% less time than the experiments without any feedback (5–6). The area explored is increased by an average of 26.5% when full feedback (1–2) is employed.

For the single planner–three explorer poor performance tests (7–12), the full feedback experiments (7–8) take on average 10.4% less time than without any feedback (11–12). In the partial failure experiments (13–16), full feedback (13–14) takes on average 19.8% less time than no feedback (15–16). A similar result has been obtained for complete failure tests (17–20), where full feedback (17–18) takes on average 19.3% less time than no feedback (19–20). In the poor performance experiments (7–12), there is no significant difference in area explored by the robots with or without feedback coordination. However, full feedback increases the overall area explored by the robots in the partial failure (13–16) and complete failure (17–20) experiments by an average of 12%.

When full feedback is employed, the single planner–three explorer experiments (7–8) take on average 35% of the time taken to explore using the single planner–single explorer configurations (1–2). In single planner–single explorer tests, task score feedback (3–4) has a larger exploration time and reduced area exploration than without feedback (5–6). However, in the single planner–three explorer tests, task score feedback (9–10) marginally reduces exploration time over no feedback (11–12).

A comparison of the number of workers employed as planner and explorers in the experiments is given in Figure 4. Generally, one to six additional explorer robots are employed when TR feedback coordination is enabled. However, it is important to note that the maximum number of explorers and planners present at any point in time during exploration is as specified in Table 1.

Figure 3 Exploration time and explored area details of test configurations (see online version for colours)

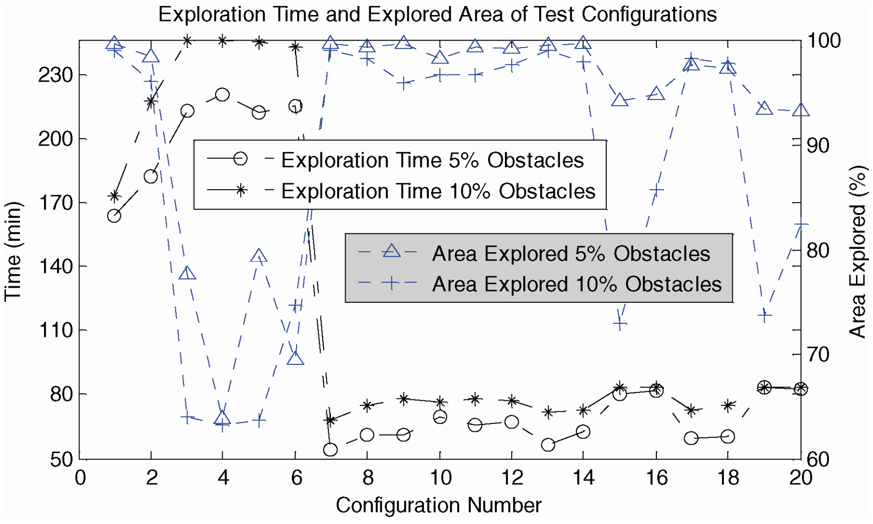
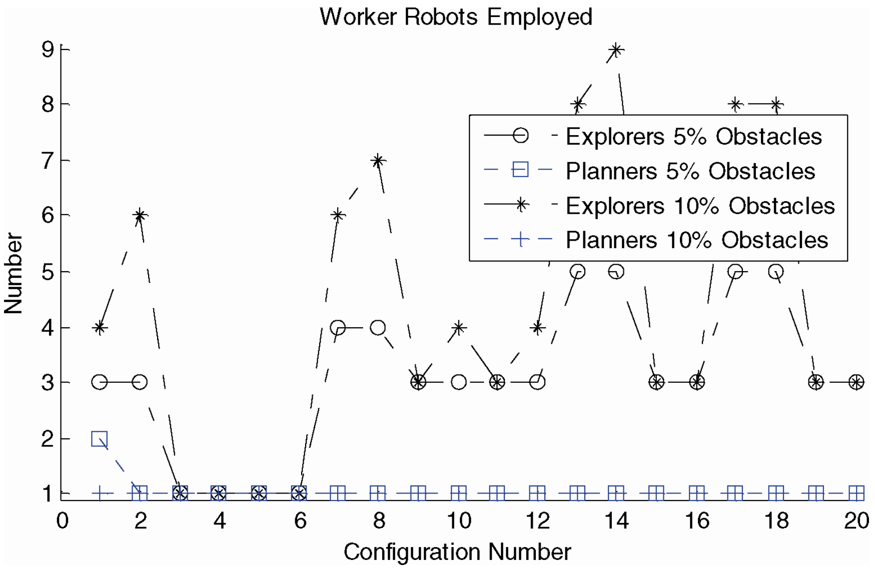


Figure 4 Worker robots employed as planners and explorers (see online version for colours)



8 Conclusion

A task allocation and feedback coordination strategy for limited capability mobile robots has been presented and applied to a multi-robot exploration task. Initial results indicate that the feedback coordination mechanism successfully detects and responds to poor performance, partial failure and complete failure of monitored robots. The mechanism can reduce exploration time by to 27.5% for a single planner–three explorer worker robot configuration. A slight increase in the area explored by the team is also achieved when the feedback mechanism is employed. Additional global worlds and worker robot configurations are being tested for inclusion at the conference. Future work includes a comparison of our method with other task allocation and coordination strategies.

References

Chand, P. and Carnegie, D.A. (2007a) 'Task allocation and coordination for limited capability mobile robots', *Australasian Conference on Robotics and Automation*, Vol. 1, Brisbane, Australia: ARAA.

Chand, P. and Carnegie, D.A. (2007b) 'Memory-time tradeoffs in a path planning approach utilising limited memory robots', *Proceedings of the International Conference on Computational Intelligence, Robotics and Autonomous Systems*, pp.243–248.

Farinelli, A., Iocchi, L. and Nardi, D. (2004) 'Multirobot systems: a classification focused on coordination', *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 34, No. 5, pp.2015–2028.

Gerkey, B.P. and Mataric, M.J. (2004) 'A formal analysis and taxonomy of task allocation in multi-robot systems', *Int. J. Robotics Research*, Vol. 23, No. 9, pp.939–954.

Kannan, B. and Parker, L.E. (2006) 'Fault-tolerance based metrics for evaluating system performance in multi-robot teams', *Performance Metrics for Intelligent Systems Workshop*, Gaithersburg, Maryland.

Kannan, B. and Parker, L.E. (2007) 'Metrics for quantifying system performance in intelligent, fault-tolerant multi-robot teams', *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.951–958.

Lee-Johnson, C.P., Chand, P. and Carnegie, D.A. (2007) 'Applications of a adaptive hierarchical mobile robot navigation system', *Australasian Conference on Robotics and Automation*, Vol. 1, Brisbane, Australia: ARAA.

Long, M.T., Murphy, R.R. and Parker, L.E. (2003) 'Distributed multi-agent diagnosis and recovery from sensor failures', *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.2506–2513.

Murphy, R.R. and Hershberger, D. (1999) 'Handling sensing failures in autonomous mobile robots', *Int. J. Robotics Research*, Vol. 18, No. 4, pp.382–400.

Parker, L.E. (1997) 'L-ALLIANCE: task-oriented multi-robot learning in behavior-based systems', *Advanced Robotics*, Vol. 11, No. 4, pp.305–322.

Parker, L.E. and Tang, F. (2006) 'Building multirobot coalitions through automated task solution synthesis', *Proceedings of the IEEE*, Vol. 94, No. 7, pp.1289–1305.

Vig, L. and Adams, J.A. (2006) 'Multi-robot coalition formation', *IEEE Transactions on Robotics*, Vol. 22, No. 4, pp.637–649.

Williamson, D.A. and Carnegie, D.A. (2006) 'Embedded platform for search and rescue applications', *Proceedings of the International Conference on Autonomous Robots and Agents*, pp.373–378.