Multi-robot Systems: A Cooperative Behaviour and Performance Evaluations for Mine Safety Inspections

Chika Yinka-Banjo
Department of Computer Science, Faculty of Science, University of Cape Town, Cape Town, South Africa
+27739544034
chika.yinka-banjo@uct.ac.za

Isaac O. Osunmakinde
School of Computing, College of Science, Engineering and Technology, University of South Africa, P.O. Box 392, UNISA 0003, Pretoria, South Africa.
+2711-670-9155
osunmio@unisa.ac.za

Antoine Bagula
Department of Computer Science, Faculty of Science, University of Western Cape, South Africa
+2721 959-2562
bbagula@uwc.ac.za

ABSTRACT

Development of cooperative behaviour for multi-robots system (MRS) is an important aspect of robotic task achievement. This is because it is safer to use more than one robot to achieve a task for continuity measurement of task completion and it is also quicker to complete a task on time with two or more robots as against one robot. Continuity measurement here means if one robot breaks down, the remaining robots can continue and complete the task. Inspection of environments such as underground tunnels, underwater and industrial pipes are harsh areas that need exploring and exploiting in field robotics. The focus of this research is on using MRS to achieve pre-entry safety inspection of roof cracks and level of toxic gases in the underground terrain with the paramount aim of saving the lives of the miners from disasters. This paper presents the model for performance evaluations in terms of scalability. The use of cooperative behaviour to address the issues of multi-robot learning is also presented. We discuss the cooperative MRS behaviour framework and then describe the results from simulations. The evaluation/scalability of the model using two robots, three robots and four robots are investigated. The results reveal that our model can guide two, three and up to four robots to achieve efficient cooperative inspection behaviour in underground terrains with minimal cost of time and memory. The model will also work very well with any number of robots so long as the environment is expanded.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Cognitive Simulation – *learning, knowledge acquisition;* I.2.9 [**Distributed Artificial Intelligence**]: Autonomous vehicles; Sensors; I.2.11 [**Robotics**]: Coherence and coordination; Intelligent Agents; Multiagent Systems.

General Terms

Algorithms, Measurement, Performance, Design, Reliability, Experimentation,

Keywords

Cooperative Behaviour, MRS, Evaluations, Performance, Underground Terrain and Safety Inspections.

SAICSIT2014, September 29 - October 01 2014, Centurion, South Africa
Copyright 2014 ACM 978-1-4503-3246-0/14/09...$15.00
http://dx.doi.org/10.1145/2664591.2664593

1. INTRODUCTION

Despite the improvements in safety in the South African mining industry, the numbers of injuries and fatalities (as shown in Figure 1), are still very high, especially those related to respiratory diseases. More prompt innovations are required to further mitigate these fatalities [1]. Figure 1 show that the fatalities are alarmingly high especially for respiratory diseases. For a full discussion of underground mines, benefits and disaster rates, see [2].

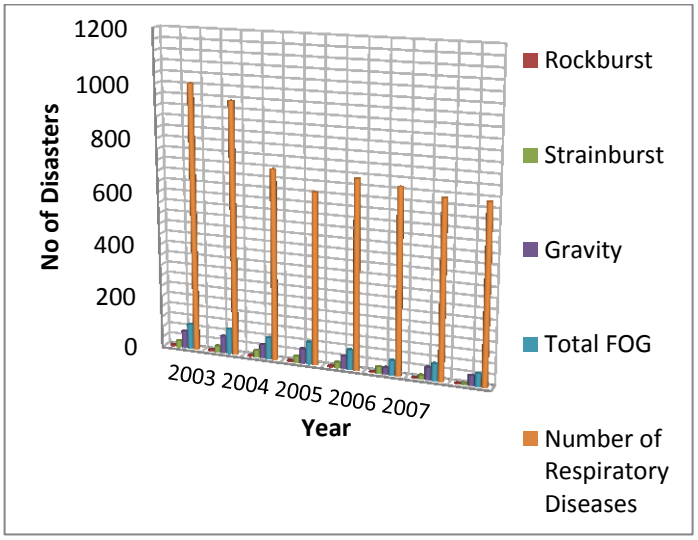

Figure 1: Fall of ground and respiratory-induced fatalities 2003 - 2010

In an attempt to proffer solutions to safety-related issues in underground mines, a multi-robot system (MRS) could play a major role. A MRS can be used to check the status of the roof and the level of toxic gases (TG) in the mine. However, a cooperative behavioural model is required for a thorough, efficient and effective inspection of underground mine safety.

The first element that will enhance the behaviour of the MRS in the mine is good communication between robots. How they interact and achieve their task cooperatively is of crucial importance. A list of literature on multiple robots' cooperation implemented in space was reviewed in [3]. Using multiple robots to achieve tasks has been more effective compared to using a single robot. See for instance [4] (and all references therein) for specific robotic tasks. This research focuses on developing a hybrid model from Q-Learning (QL) and ant colony optimization (ACO) algorithms that will guide robots in achieving cooperative behaviour in the mine. The model that has been developed is tested and verified using numerous robots. Figure 2 depicts three cooperating robots replacing human beings and taking action in the mine.

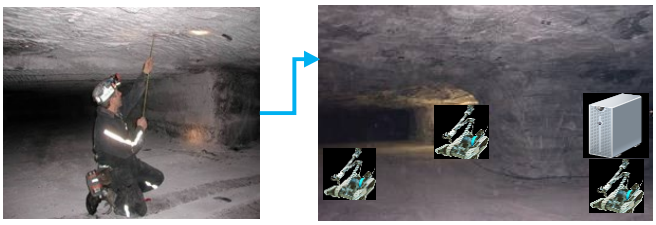

Figure 2: Overview of robots supporting humans and performing cooperative inspection in an underground mine

The communicating robots are used to inspect the status of the mine roof and the level of TG in the mine as a safety routine check before miners are deployed for operations. The result of the inspection specifies how safe the mine environment is for operations. A major focus in this paper is to build on the model developed (QLACS) and determines how many robots can use the model to achieve mine safety inspection under a time constraint. The major contributions of this paper are as follows:

- Application of the developed hybrid QLACS model to scalability of two, three and four robots for addressing cooperative behaviours in MRS and achieving underground terrain safety inspections.
- Performance evaluations of time and memory scalability on the hybrid model, and comparative analysis on cooperative behaviour systems.

2. BACKGROUND STUDIES

Related work on the safety of underground and other terrains for MRS are described in this section. The fundamental theory of the reinforcement learning algorithm is established and the underlying principles of ACO are demonstrated.

2.1 Related MRS for Underground Terrains

Safety is a predominant element in an underground mine [5]. Large rock falls, underground fires, premature initiation of explosives, gas inhalation etc. are typical underground mine misfortunes. To contribute to underground mine safety, that is, to prevent disasters from occurring, interacting autonomous multi-robots are sent into the mine before operations resume to determine how safe the underground mine is. This they achieved by assessing the status of the rocks and roofs and the level of TG.

The automation of platforms such as air, sea, and land in harsh unstructured environment has been areas of interest in field robotics [6]. Although field robotics has its own challenges, researchers are beginning to make progress in these harsh environments that are cooperatively constrained.

The proposed cooperative behavioural model presented in [2] promised to handle the safety part of field robots presented by [6] in underground mines. Their model architecture has three layers; the first layer handles the cooperative behaviour of the model, the second layer deals with the scalability degree of the model while the last layer handles the applicability of the model. This paper is building on what has already been proposed in [2] by evaluating the second layer of the model architecture.

Parker et al. [7] identify a need in MRS to develop a mechanism that would enable robot teams to generate cooperative behaviours autonomously. The cooperative multi-robot observation of multiple moving targets application was presented as a rich domain for studying the issues of multi-robot learning of new behaviours. The need for learning and adaptation was identified and revealed from their implementation results.

The need to extend cooperative behaviour of MRS to underground terrains has been mentioned. Dangerous safety inspection of the mine by humans immediately after blasting can be replaced by inspection by multi-robots. This will reduce the dangers faced by miners and mine inspectors.

While mine safety rules and training have been used to guide underground miners in past and recent times, the use of autonomous robots is gradually being introduced [8]. However, a robust model that can guide MRS to achieve a safe environment for miners before mining operations resume is yet to emerge. Thus, it is an ongoing research challenge. Our model is built by hybridizing two artificial intelligence algorithms called QL and ACO.

2.2 Studies on Reinforcement Learning

Reinforcement Learning is learning by interaction with an environment. In the last decade, a class of approach in which the agent learns based on reward and punishment it receives from the environment, called reinforcement learning (RL), has become the methodology of choice for learning in a variety of domains, especially the robotic domain [9]. RL is one of the artificial intelligence algorithms that can achieve learning by experience. This enhances robots' interaction with the environment.

RL problems are made up of four elements: a set of states; a set of actions for each state; a transition function which specifies the probability of transitioning from one state to another when performing each action; and a reward function, which indicates the amount of reward or cost associated with each transition [10].

The use of RL is investigated to assist the behaviour of an MRS in safety inspection of underground mines. At any time step, each robot is in a specific state in relation to the environment and can take one of the following actions: inspect, ignore or shut down. Each robot receives feedback after performing an action, which explains the impact of the action in relation to achieving the goal. The effect of the action can be either a good or bad reward. This reward is measured in terms of values. Therefore, the value of taking an action a in any state s of the underground terrain is measured using the Action-Value function called Qvalue $Q^{\pi}(s,a)$. When a robot is starting from state s, taking action a, and using a policy π, an expected return, which is defined as the sum of the discounted rewards, is achieved.

Table 1: Q-Learning Algorithm

Steps:
initialize $Q(s,a)$ *arbitrarily*
repeat (for each episode):
initialize s
Repeat (for each step of episode):
Choose a *from* s *using policy derived from* Q
Take action a *, observe* s, s'
$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
$s \leftarrow s'$
until s *is terminal*

For the purpose of this research, the QL algorithm, a method of RL, is used to explore $Q^{\pi}(s,a)$ values so as to achieve optimal actions in the states. Table 1 explains QL used in developing one of the components of the QLACS model.

α is the learning rate set between 0 and 1. At 0, Q-values are never updated, hence nothing is learned; learning can occur quickly at 1. γ is the discount rate set between 0 and 1 as well. This models the fact that the future rewards are worth less than the immediate rewards. maxa' is the maximum reward that is attainable in the state following the current state. That means the reward for taking the optimal action thereafter.

The QL architecture developed in this research consists of learning threads which amount to the number of robots involved in the behavioural inspection tasks. Each robot in the learning thread carries out QL in the environment. The Q values are checked for and compared with zero to determine when each robot takes an action of inspect, ignore or shutdown. The improved QL explains how the QL algorithm that is adopted for the cooperative part of the work is designed. This also enhanced better communication with the robots.

2.3 Studies on Ant Colony Optimization

Ant Colony Optimization is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. It is one of the most successful techniques of swarm intelligence used in solving real life problems. The use of ACO for solving several kinds of problems based on the social insect metaphor has attracted increasing attention in the artificial intelligence community [11]. There are different variants of ACO [12, 13]. Dorigo et al. propose the ant colony system (ACS), a variant of ACO, for combinatorial optimization based on the observation of ant colonies' behaviour [14]. ACS is derived from ant system (AS). The AS is the first ACO algorithm proposed in the literature by [15]. In AS, an ant k being in the node i chooses the next node j with a probability given by the random proportional rule defined as

$$P(i,j) = \frac{[\tau_{i,j}]^{\alpha}\,[\eta_{i,j}]^{\beta}}{\Sigma_{n\in N^k}[\tau_{i,h}]^{\alpha}\,[\eta_{i,h}]^{\beta}} \qquad (1)$$

where N^k is its feasible neighborhood. Parameters α and β in Equation 1 have the following influence on the algorithm behavior. If α = 0, the selection probabilities are proportional to $[\eta_{i,j}]^{\beta}$ and the closest node will more likely be selected: in this case AS corresponds to a classical stochastic greedy algorithm (with multiple starting points since ants are initially randomly distributed on the cities). If β = 0, only pheromone amplification is at work: this will lead to the rapid emergence of a stagnation situation with the corresponding generation of tours which, in general, are strongly suboptimal. Search stagnation is defined in as the situation where all the ants follow the same path and construct the same solution. The solution construction ends after each ant has completed a tour, that is, after each ant has constructed a sequence of length n. Next, the pheromone trails are updated. In ACS, construction of a solution uses the pseudo-random proportional rule. This means that the most attractive edge is selected greedily with fixed probability.

3. PROPOSED FRAMEWORK FOR COOPERATIVE MRS BEHAVIOURS

This section details the framework of the proposed model, the problem formulation, navigation and communication behaviours and the development of the new hybrid model. This work developed a cooperative behavioural model that will guide an autonomous multi-robot system into achieving an inspection task in the underground terrain. Figure 3 depicts the proposed hybrid model framework. The framework is divided into three layers: the cooperative behavioural layer (bottom layer), the scalability layer (middle layer) and the application layer (topmost layer).

The cooperative and route-finding behaviours of the multi-robots are handled in the bottom layer. In this layer, two robots learn and adjust to the environment with the help of an intelligent hybrid model designed using QL and ACS algorithm. Their cooperative behaviours are tested with the model in the layer. Then, with an increment in the size of the robots, the scalability features of the model are verified at the middle layer. Both the bottom and the middle layers are tested for memory usage and time consumption in the course of the experiment. The results acquired from the bottom and middle layers are stored for future use in the topmost layer.

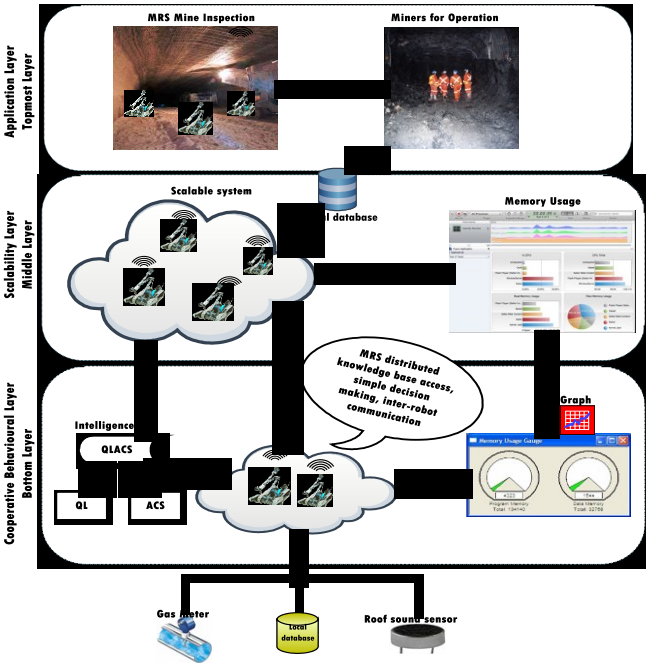

Figure 3: Framework of the proposed QLACS model for cooperative behaviours

3.1.Problem Formulation

Suppose we have seven rooms/states connected by doors/links representing underground mine regions, as shown in Figure 4 and labelled as shown in Table 2. We label the rooms A to F. The outside of the mine can be thought of as one big room (H). Doors F and C lead outside the mine, H. We put two robots in rooms F and C as their starting states respectively. The robots inspect one

state at a time, considering the obstacles encountered in the process. The robots can change direction freely, having links/doors to other rooms/states. In each of the states in Figure 4 two actions are possible: inspection of roof cracks (RC) and level of TG, or ignoring the state, as it has been inspected earlier. According to our proposed model, if a robot can inspect at least half of the given underground mine region to attain its maximum performance, it will attract a good reward. When the current state of a robot is the end point of the inspection task, the robots exit the mine using the exit points C and F respectively.

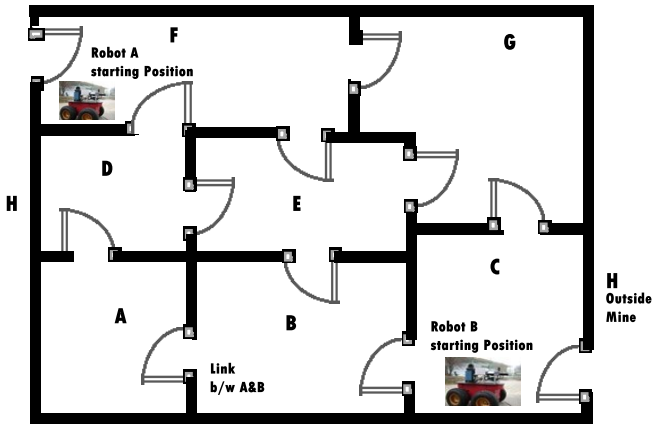

Figure 4: Model of the Environment with Two Entrances and Exits (2EE)

Table 2: States and possible actions of the environment

	State	Possible Actions
1	A (Lower Left Part (LLP))	Inspect, Ignore
2	B (Lower Middle Part (LMP))	Inspect, Ignore
3	C (Lower Right Part (LRP))	Inspect, Ignore
4	D (Middle Left Part (MLP))	Inspect, Ignore
5	E (Central Part (MCP))	Inspect, Ignore
6	F (Upper Left Part (ULP))	Inspect, Ignore
7	G (Upper Right Part (URP))	Inspect, Ignore
8	H (Outside Mine Part(OMP))	Shutdown

3.2.Basic Navigation and Cooperative Behaviours using QLACS

In developing the navigation and cooperative behaviours of QLACS, we consider the transitions states and state diagrams. The possible transition states of the robots are displayed using the state diagram in Figure 5. The state diagram and the transition matrix in Table 3 are formed using the QL algorithm. The possible actions were set for each robot as inspect, ignore and shutdown (after reaching the goal state H). Also, a reward system that will reflect the possible actions of the robots was chosen as illustrated in Table 3. In other words, a robot gets 150 points only when the goal is achieved (shutdown), 100 points for ignoring an already inspected area (ignore) and 50 points for inspecting an uninspected area (inspect). The allocation of values for the reward system could be subjective. Two methods were used in investigating the navigation behaviours of the MRS in this research. One of the methods was used in addressing the navigation and cooperative behaviours of the MRS and the other method was used only to address the navigation behaviours of the MRS. The second method was explored because the first method failed to give an optimized route but an efficient communication system was nevertheless achieved.

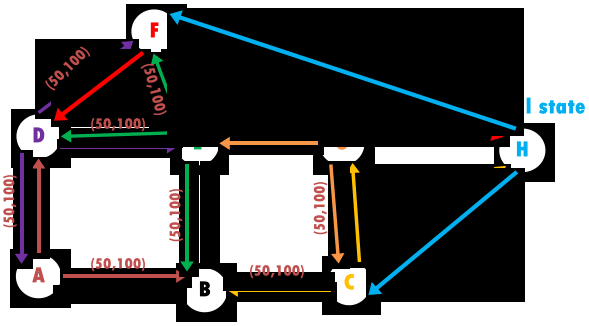

Figure 5: Events and Transition Diagram of the Modelled Environment with H as Goal State

Table 3: Initial Reward Matrix

Robot's State	Robot's Action							
	A	B	C	D	E	F	G	H
A	-	50, 100	-	50, 100	-	-	-	-
B	50, 100	-	50, 100	-	50, 100	-	-	-
C	-	50, 100	-	-	-	-	50, 100	150
D	50, 100	-	-	-	50, 100	50, 100	-	-
E	-	50, 100	-	50, 100	-	50, 100	50, 100	-
F	-	-	-	50, 100	50, 100	-	50, 100	150
G	-	-	50, 100	-	50, 100	50, 100	-	-
H	-	-	50, 100	-	-	50, 100	-	-

The second method, on the other hand, gave an optimized route for the inspection task. This led us to harness the strengths of the two methods into a hybrid method. The new hybrid method forms our new cooperative behavioural model (QLACS). QLACS is formed by two improved algorithms; ACS is the first component and QL the second component. Some heuristics were added to the traditional ACS and QL to achieve the hybrid QLACS. The second component of QLACS, which is an improved QL, was initially used to solve the proposed problem. After much analysis, we realized that the system needs to be optimized for effective navigation behaviour. The random method used to determine which direction the robots navigate to, called RandomStateSelector, did not guarantee the shortest possible time for inspection because of repeated states decisions. However, the method CheckInspection, which is responsible for broadcasting among the robots, worked efficiently. The communication strength of this method was harnessed and we use the first component of QLACS to address the navigation issue.

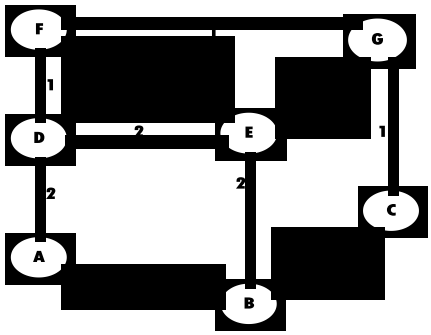

Figure 6: Weighted map/graph of the model environment

Table 4: Combined adjacency and weight matrix

	F	G	E	D	A	B	C	H
F	0	1	1	1	0	0	0	1
G	1	0	2	0	0	0	1	0
E	1	2	0	2	0	2	0	0
D	1	0	2	0	2	0	0	0
A	0	0	0	2	0	2	0	0
B	0	0	2	0	2	0	1	0
C	0	1	0	0	0	1	0	1
H	1	0	0	0	0	0	1	0

The random selector problem identified in QLACS second component implementation was handled as indicated in Figures 6 and Table 4, using the route-finding algorithm. The number of obstacles the ant encounters forms the basis of the weighted graph in Figure 6. The combined table in Table 4 contains the weights of the obstacles and evidence of an edge between any two vertices (states). It shows that there is a connection between any two vertices in a graph. Generally, a "1" or "2" depicts the existence of an edge while a "0" represents the absence of an edge, i.e. no transition between such vertices. The constructed graph is an undirected multi-graph, providing evidence of some agents coming from F or C of the mine (logical space). It is unidirectional because agents can move in any particular direction (multi-graph). This means that the same weights on the edges apply to both directions. The graph does not show H; we assume that once the agents reach F or C, they exit if all inspections have been done.

3.3.Development of Hybrid Model

Our cooperative behavioural model (hybrid model) is an integration of two algorithms: (1) a route-finding algorithm and (2) communication and cooperative algorithm. The integration works this way: the optimal route finder (ACS) determines where the robot goes from the starting point to the destination, while QL determines what action it takes when it gets to any of the states. This collaboration works effectively because the optimal route finder has been proven to give the best possible transitions and the heuristically accelerated QL has proven to be effective by showing evidence of effective communication in making inspection decisions. Consequently, it guarantees the shortest possible time for inspection in the absence of wasteful inspection decisions. The analytical development of the hybrid model is described below.

When ACS starts, the edge attractiveness is computed as follows:

$$\eta_{i,j} = \frac{1}{D_{i,j}} \quad (2)$$

Computation of instantaneous pheromone by ant k

$$\Delta\tau^k = \frac{Q}{L_k} \quad (3)$$

Update of pheromone

$$\tau_{i,j} = (1-\rho) * \tau_{i,j} + \Delta\tau^k{}_{i,j} \quad (4)$$

Computation of edge probability

$$P_r(i,j) = \frac{[\tau_{i,j}]^\alpha [\eta_{i,j}]^\beta}{\Sigma_{e'=(i,j)}[\tau_{i,j}]^\alpha [\eta_{i,j}]^\beta} \quad (5)$$

Adoption of roulette wheel

$$\text{Cumulative}(P_r(i,j)) = \sum_{i=1}^{N+1} P_r(i,j) \quad (6)$$

$$f_i = \frac{\sum_{j=1}^{N} f_j}{N} \quad (7)$$

$$P_i = \frac{f_i}{\sum_{j=1}^{N} f_j} \quad (8)$$

Where α is the pheromone influence factor, β is the influence of adjacent node distance, ρ is the pheromone evaporation coefficient, Q is attractiveness constant, e is the visited edge, e' is edge not visited , Lk is the length of the tour of ant k, τ is the pheromone concentration (amount), $\boldsymbol{\eta}$ is the specific visibility function (attractiveness), $\Delta\tau^k$ is the pheromone concentration by the Kth ant, $P_r(i,j)$ is probability of moving from i to j, $D_{i,j}$ is visibility or distance between i and j, f_i is fitness of individual in a population, P_i is the probability of being selected among f_i, N is the number of individuals in the population, i,j denotes any two adjacent nodes in the graph. Equations 2 to 8 build the complete route-finding model. Equations 2 to 4 are prerequisite to equation 5 while Equation 5 is prerequisite to roulette wheel selection. At the end of equation 8, new states are selected and the trail is updated. The best path from both directions is selected and used as input in QL.

When QL starts, each robot in its QL thread and its learning rate is computed as follows:

$$\gamma = \frac{0.5}{[\,1+Frequency\,(s,a)]} \quad (9)$$

Q-values are updated

$$Q(s,a) = R(s,a) + \gamma(\max(Q'(s,a)) \quad (10)$$

Making a broadcast (Decision = Inspect/Ignore/Shutdown)

$$Q(s,a) = \begin{cases} Q = 0 \;\; Inspect\ if\ s_j \neq goalstate \\ Q > 0 \;\; Ignore\ if\ s_j \neq goalstate \\ Q \geq 0 \;\; Shutdown\ if\ s_j = goalstate \end{cases} \quad (11)$$

Where Q is the Qvalue update, s is the state, a is action, R is reward, γ is the learning rate. Equation 11 marks the end of the hybrid model development. Equations 9 to 11 are state-dependent. The states are kept in a buffer and then accessed at run time. ACS and QL do not work simultaneously. ACS works to completion and QL takes the final output as its input. ACS is not repeatedly called while QL is working. Figure 7 gives the details of the new hybrid algorithm.

```
INPUT: Edge distance(obstacles), pheromones,
ants' trail, associated probabilities,
 starting and terminating indexes i.e. from F or C
OUTPUT: Effective cooperation, inspection and
navigation
1.  Boolean CompletedFlag = False //Boolean
    variable indicates completion for all the threads
2.  Declare CompletedThreadBuffer // Data structure
    stores Info about completed thread
3.  Initialize all Qvalues to Zero      //All Qvalues
    positions are initialized to zero
4.  Initialize Best Paths From ACO algorithm  //
    starting from F and C
5.  While (CompletedFlag <> True)//Checks for when
    all robots have finished and flag is true
    Begin
      Create N number of Threads in Parallel
      Threads get Current States in Parallel from ACO
    algorithm
      Threads get Next States and Indexes in Parallel
    from ACO algorithm
      Threads compare Qvalues of all corresponding
    positions of Current States (Each Robot Broadcast
    Qvalue info)
      IF ((Q ==0) & (ThreadNextState <>
    GoalState))//Checks if a particulate state is
    available
        Begin
          State is Available, Robot with the
    CurrentThreadID Inspects
          Compute and Update Qvalue
      End
    IF (Q >0) // checks if a state is not available,
    because an already inspected state has Q>0
      Begin
         State is already inspected, Robot with the
    CurrentThreadID Ignore
        Compute and Update Qvalue
     End
    IF ((Q ==0) & (ThreadNextState == GoalState)) //
    Checks for goal state and shuts down.
      Begin
       Compute and Update Qvalue
       Goal state is reached and Inspection Completed
       Thread with the CurrentThreadID Shuts down
       Store CurrentThreadID in CompletedThreadBuffer
     End
    IF (Count [CompletedThreadBuffer] ==
    NumberOfRobot)//Learning stops when this
    happens
     Begin
       CompletedFlag= True
    End
End of While Loop.
```

Figure 7: Pseudo-code for QLACS

4. SCALABILITY OF QLACS MODEL: PERFORMANCE EVALUATIONS

Building on the result obtained using two robots, the qualitative and quantitative experimental evaluation approaches are considered in this section. The main focus of this work is using multi-robots to achieve cooperative inspection tasks in underground terrains. The implementation designs and the class diagrams showing interactions of objects are explained in 4.1. For the experiments conducted in this work, different test of the developed model are performed in 4.2, 4.3 and 4.4 using two, three and four robots respectively. The comparative analysis on cooperative behaviour system is detailed in 4.5.

The Pseudo code in Figure 7 was translated and coded into C# source code. The simulation was carried out using a computer with Intel core i3 and 4GB of RAM. Various simulation results were then recorded based on the evaluation criteria.

4.1.Experimental Setup

In this section, we use an underground scenario to explain the experimental setup and the class diagram to illustrate the communication of the classes. see Figures 8 and 9. The scenario in Figure 8 shows the underground terrain with two communicating robots exhibiting inspection behaviour. The environment is partitioned into seven different rooms. The first robot R1 takes action from state F to state E and communicates its action to robot R2. Robot R2 then takes action from state C to state B and communicates to robot R1. This process is continued until the environment has been fully inspected and each robot exits the terrain via the closest exit point to H.

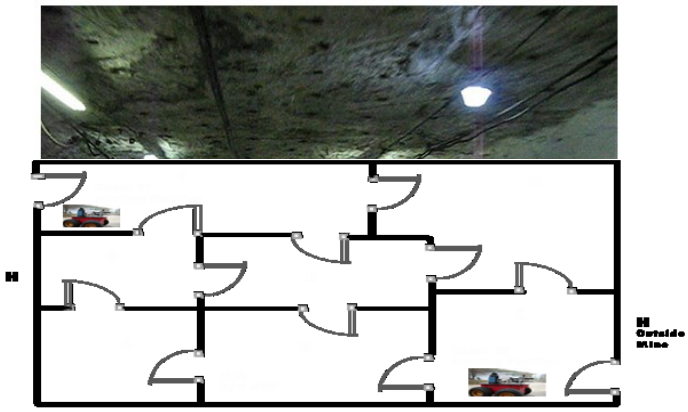

Figure 8: Underground terrain with segmented regions and two robots

The proposed model for the cooperating behaviour of autonomous robots has five integrating classes: the ACS, thread library, sensor, QL and graph classes. The ACS class does the route finding and it is the genesis class that must provide two optimized paths for the robots coming from the two entrances or exits of the mines. We refer to this model of the environment as two-exit and entrance (2EE). These optimized routes are relayed to the QL class during inspection. The QL class has a direct association with the thread library class. The thread library class is also associated with the ACS class. The QL class is supposed to make the robots learn which decision is best when the sensor class renders environmental readings of the mine. The sensor readings could indicate a crack in the roof of the mine (RF) or level of gaseous toxicity in the mine environment (TG). The sensor class manages the range of values as perceived from the sensors on the autonomous robots. The decisions of the robots are determined by the modified accelerated QL algorithm, which could be [Inspect], [Ignore] or [Shutdown]. The QL class communicates with the sensor class while the agent is passing through each position in the mine, because the inspection must be complete and thorough. The graph class is the last class that picks up accumulated values from the learning class and populate the chart control based on the customized context of parameters we want to use in measuring the

overall performance. Figure 9 is the unified modelling language (UML) pictorial view of the classes.

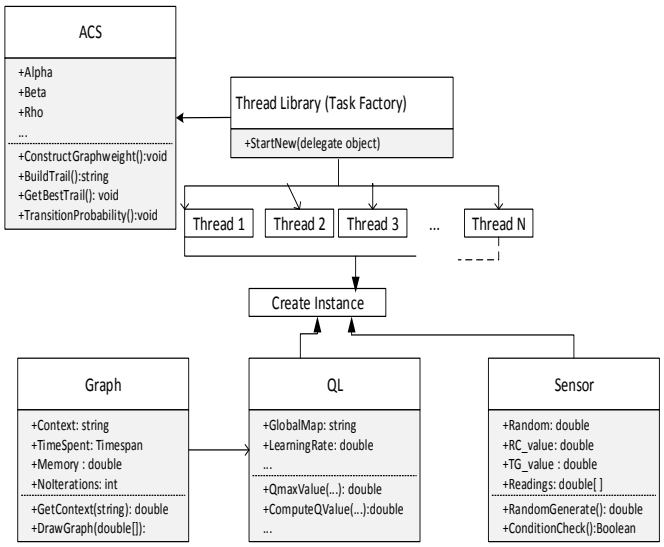

Figure 9: Class diagrams showing interactions of objects

It shows the key building blocks that enable designers to represent a given system's classes, their attributes and functions that are associated with them and the relationship among the different classes that make up the system. Both ACS and QL are associated directly to the thread library. We rely on the thread library (task factory) in Microsoft DotNet libray to create several instances of threads in parallel, may be one to N number of threads. We used this thread library because from the available documentation it is adjudged the best and provides a thread safe environment. It also has the advantage of passing in delegates and methods that returns value unlike other libraries that do not allow values to be returned. This makes it easier to access and manipulate individual Qvalues and sensor reading in its own thread without race condition or deadlock and also makes the thread library to be thread safe.

Each thread creates an instance of the Qlearning class and sensor class. The sensor class simulates or manages readings from the hardware devices. The sensor class is to detect two conditions: roof crack (RC_Value) and toxic gas level (TG_Value). The Qlearning class learns based on these values to determine what actions are to be carried out. The Qlearning class supports communications among the competing threads i.e. they share intelligence of states they had visited. Through this communication the threads are aware of other robots thereby avoiding multiple inspections with reduced inspection time.

Through the capability in the thread library, each thread has access to the optimal path (from ACS). When a thread gets to a state, it calls the sensor class to deliver its readings to the Qlearning class then decisions are taken. The Qlearning class uses the *Display* method to show all results of learning by each thread.

The graph class relies on the parameters supplied and Qvalues generated during the exercise to plot a chart. The charts are based on the selected context of performance. The context of performance refers to the performance yardsticks like: memory usage, inspection time, states inspected by each robot against the number of robots etc.

The notations on Figure 9 have the following meaning:

- It shows that at least one sensor is needed for one robot to sense the environment.
- At least one robot with QL thread is needed to do a complete inspection.
- The graph gives details of at least one learning robot.

In summary, we use at least two robots with two QL threads to achieve our cooperative behavioural model development.

4.2.Experiment 1: Performance of QLACS with two robots-based MRS

Some results obtained by experimenting with the cooperative behavioural action of two robots using the QLACS model are presented in this section. The performance of the two robots was evaluated by running the simulation 10 times, using the same number of agents. The performance of the proposed QLACS model proposed shows good communication between the two robots as displayed in the last two columns of Table 5. The time and memory used in achieving these inspections are recorded in columns 2 and 3 of Table 5. Having achieved good cooperation between two robots, Figure 11 depicts the simulated movement of the two robots. The blue long broken lines depict the movement of robot R1 while the red short broken lines depict robot R2's movement. The blue double arrows show the states inspected by robot R1 and the red double arrow lines show the states inspected by robot R2. The flow of movement and the states visited by each robot according to Table 5 and Figure 10 create accurate understanding of the cooperation between the two robots.

Table 5: Performance of QLACS with two robots-based MRS

No of runs	Time (sec)	Memory Usage (bytes)	Inspected states (R1)	Inspected states (R2)
1	7.0004	10304	F,G,E,D	C,B,A
2	10.0006	10300	F,G,E,D	C,B,A
3	10.0006	10264	F,G,E,D	C,B,A
4	11.0007	10296	F,G,E,D	C,B,A
5	11.0006	10296	F,G,E,D	C,B,A
6	8.0005	10288	F,G,E,D	C,B,A
7	7.0004	10288	F,G,E,D	C,B,A
8	7.0004	10256	F,G,E,D	C,B,A
9	7.0004	10836	F,G,E,D	C,B,A
10	8.0004	10288	F,G,E,D	C,B,A

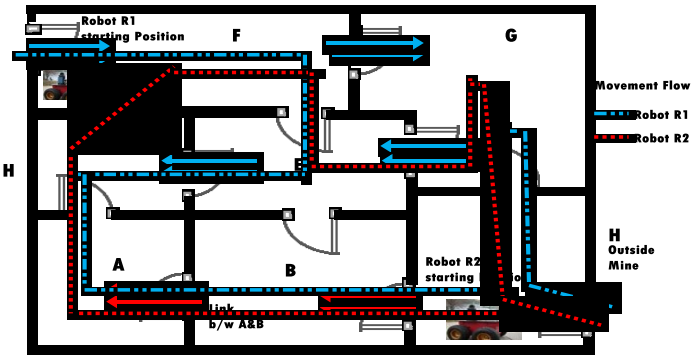

Figure 10: Movement of two robots showing cooperation

4.3.Experiment 2: Performance of QLACS With three robots-based MRS

In investigating the scalability of the QLACS model, three robots are used in an experiment on cooperative inspection of the underground terrain. The simulated environment is run 10 times just as when using the two robots. The performance obtained from the team of three robots was compared to the performance from the team of two robots; although there is slight increase in time and memory usage, the three robots achieve good communication and effective inspection, as shown in Table 6, because there are no repetitive inspections. The flow of movement and the states inspected by each robot are depicted in the simulated movement of the three robots in Figure 11. For the entry points of the three robots, as shown in Figure 11, we use the result obtained from the optimized route-finding component of the QLACS model. Robot R1 enters the terrain through state F and exits through state C, robots R2 and R3 enter through state C and exit through state C. Results show that our model QLACS can handle three cooperating robots conveniently with almost stable time and memory usage.

Table 6: Performance of QLACS with 3 Robots-Based MRS

No of runs	Time (sec)	Memory Usage (bytes)	Inspected states (R1)	Inspected states (R2)	Inspected states (R3)
1	8.0005	13840	C	F,E,D	B,A,G
2	23.0013	13850	C	F,E,D	B,A,G
3	8.0005	13840	C	F,E,D	B,A,G
4	7.0004	13840	C	F,E,D	B,A,G
5	11.0006	13852	C	F,E,D	B,A,G
6	8.0004	13840	C	F,E,D	B,A,G
7	16.0009	13852	C	F,E,D	B,A,G
8	17.001	13840	C	F,E,D	B,A,G
9	17.001	14936	C	F,E,D	B,A,G
10	12.0006	13852	C	F,E,D	B,A,G

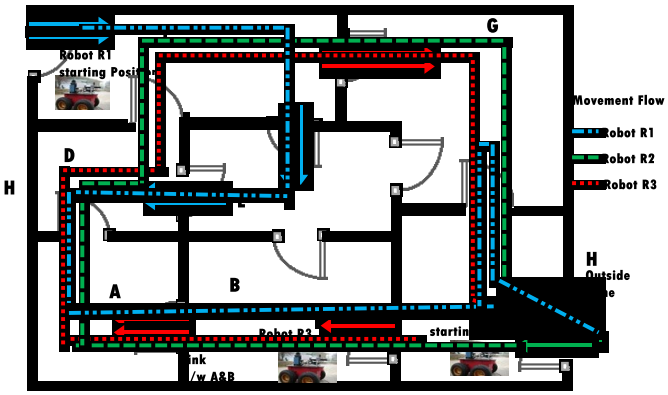

Figure 11: Movement of three robots showing cooperation

4.4.Experiment 3: Performance of QLACS with four robots-based MRS

To investigate the scalability of the QLACS model further, four autonomous robots' performance obtained by running the simulated model 10 times was analysed. It is interesting to note that the results show good cooperation between the four robots and that there is not much difference in the time used in achieving the inspection for four and three robots. Though the memory increased slightly, more than for three robots, stable memory usage was maintained. The detail of the simulation result is laid out in Table 7. Figure 12 shows the simulated movement of the four robots. Robots R1 and R3 enter the mine through state F and exit the mine through state C. The blue and purple arrows signify states inspected by robots R1 and R3 respectively. Robots R2 and R4 enter the mine through state C and exit through state C. States inspected by robots R2 and R4 are shown by green and red double arrows respectively. The legend on Figure 12 show blue line for R1 and R3 and green line for R2 and R4. This is because robot 1 and robot 3 passed through the same route. The same goes for robot 2 and robot 4. Marking the paths for robot 2 and robot 4 will be repeating the same paths and therefore making the drawing clumsy.

Table 7: Performance of QLACS with four robots-based MRS

No of runs	Time (sec)	Memory Usage (bytes)	Inspected states (R1)	Inspected states (R2)	Inspected states (R3)	Inspected states (R4)
1	10.0006	18320	F	C	G,E,D	B,A
2	14.0008	18320	F	C	G,E,D	B,A
3	10.0006	18320	F	C	G,E,D	B,A
4	11.0006	18868	F	C	G,E,D	B,A
5	8.0005	18312	F	C	G,E,D	B,A
6	10.0005	18320	F	C	G,E,D	B,A
7	11.0006	10320	F	C	G,E,D	B,A
8	10.0006	18320	F	C	G,E,D	B,A
9	7.0004	18200	F	C	G,E,D	B,A
10	10.0006	19408	F	C	G,E,D	B,A

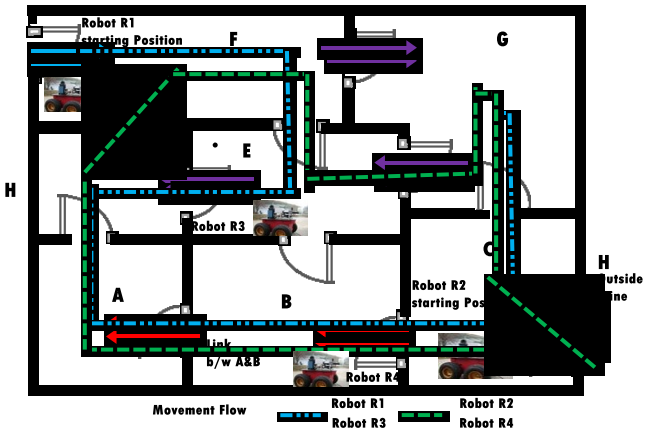

Figure 12: Movement of four robots showing cooperation

A notable observation emanating from a comparison of Tables 6 and 7 is that there is a need for a larger mine area of inspection because robot R1 in Table 4 could inspect only state C while robots R1 and R2 in Table 7 could inspect only F and C respectively. This implies that the size of the field of inspection is proportional to the number of robots to be deployed.

However, with regards to time and memory, the trends for two, three, and four robots from Tables 5, 6 and 7 show that our proposed hybrid model QLACS is scalable in terms of the number of robots that can be used to achieve the inspection task under good timing and stable memory usage. The model promises to handle a reasonable number of robots even when the environment is expanded.

4.5 Comparative Analysis on Cooperative Behaviour Systems

By investigating the similarities and differences of our cooperative behavioural model with other existing cooperative models, we use the criteria displayed in Table 8. Looking at the table, we can see from the features column that the four different models presented have different architectures, domains, intelligence, navigation approaches, conflict resolution methods and cooperation strategies. However, not all four methods have a scalability feature. This paper has shown how scalable our model is by experimenting with the performance of two, three and four robots.

5. CONCLUDING REMARKS

This work has demonstrated the usefulness of enhancing the underground terrain pre-safety inspection with multi-robots as against humans. Investigation of the size of robots is used to demonstrate how scalable the proposed model is. The QLACS model is used as the intelligence that will help the robots to cope with the dynamic environment, find the optimal cooperation strategy and make the entire system flexible.

We conducted an experiment on different numbers of robots to prove the scalability of the system. The first experiment involved two robots using the QLACS model. The performance proved good communication and cooperation between them. We also conducted other experiments on the QLACS model using three and four robots. They both revealed good communication and cooperation among the robots but needed an expansion of the mine environment to exhibit the full potential of the QLACS model in handling a different number of robots.

The efficiency of the cooperative behaviour is evaluated by a scaling relation between the task completion time, memory usage and the number of robots. The results as shown in Tables 5, 6 and 7 prove to enhance the cooperative behaviour of a team of robots. This will accelerate the rate of work output for MRS in the underground terrain.

The result of this work is an essential application that will reinforce multi-robot cooperative behaviours in any underground terrain task and in field robotics in general.

For future work, more parameters, such as the size of the robot, the camera and its focal length that could affect cooperative behaviour will be considered when upgrading the proposed model. The results could also be improved if each thread of robot is attached to a processor on duo-core laptops, high performance computing or clusters computers and both theory and experiments need to be expanded.

Table 8: Comparative Evaluation of Existing Cooperative Behavioural System with our Proposed System

S/N	Features	Action Selection Model[16]	Pheromone Route-finding Model[17]	Machine-learning Model[18]	Proposed QLACS Inspection Model
1	**Domain**	Robot soccer platform	Openstreetmap-based network layout	Object transportation task	Underground terrain safety inspection
2	**Scalability by Number of Robots**	Not scalable	Scalable	Not scalable	Scalable with time, memory and inspection
3	**Intelligence Methods**	Modular Q-learning	Cooperative ACO algorithm	Integrated RL and genetic algorithm(GA)	Integrated QL and ACS (QLACS)
4	**Conflict Resolution**	Coupled agent is proposed to resolve conflicts	Pheromone-related	Sequential QL algorithm	Broadcast and communication
5	**Cooperation Strategy**	Uni-vector field following	Route guidance through cooperative pheromones	Sequential QL	Information sharing

REFERENCES

[1] Chamber of Mines of South Africa. Fact & Figures 2013. Available online: http://www.bullion.org.za/content/?pid=71&pagename=Facts+and+Figures (accessed on 13 April, 2014).

[2] Yinka-Banjo, C.O.; Osunmakinde, I.O.; Bagula, A., Autonomous Multi-robot Behaviours for Safety Inspection under the Constraints of Underground Mine Terrains, Ubiquitous Computing and Communication Journal; ISSN 1992-8424,(7) 5,pp 1316 – 1328, 2012.b

[3] Leitner, J., A Survey of Multi-robot Cooperation in Space, Advanced Technologies for Enhanced Quality of Life, IEEE Computer Society, Vol. 37, pp. 144-151, 2009.

[4] Dudek, D. Jenkin, M. Milios, E. Wilkes, D., A Taxonomy for Multi-agent Robotics, Journal of Autonomous Robots, Vol. 3, pp. 375 – 397, 1996.

[5] Yarkan, S. Guzelgoz, S. Arslan, H. Murphy, R.R., Underground Mine Communications: A Survey, in IEEE Communication Surveys & Tutorials, pages 125 – 142, Vol. 11, 2009.

[6] Thorp, C. Durrant-Whyte, H., Field Robots, in Tenth International Symposium on Robotics Research, Vol. 6, pp. 329 – 240, 2003.

[7] Parker, L.E, Touzet, C., Multi-robot Learning in a Cooperative Observation Task, Distributed Autonomous Robotic Systems 4, ISBN: 978-4-431-67919-6, pp. 391 – 401, 2000.

[8] Teleka S.R. Green J. J. Brink S. Sheer J. Hlophe K., The Automation of the 'Making Safe' Process in South African Hard-rock Underground Mines, International Journal of Engineering and Advanced Technology (IJEAT) Vol. 1, pp. 1-7., April 2012..

[9] Mataric, M.J., Reinforcement Learning in Multi-robot Domain, Autonomous Robots, Vol. 4, pp. 73 – 83, 1997.

[10] Botvinick, M.M. Niv, Y. Barto, A.C., Hierarchically Organized Behaviour and its Neural Foundations: A Reinforcement Learning Perspective, Cognition, Vol. 113, pp. 262 – 280, 2009.

[11] Bianchi, R.A.C Ribeiro, C.H.C. Costa, A.H.R, On the Relation between Ant Colony Optimization and Heuristically Accelerated Reinforcement Learning, 1st International Workshop on Hybrid Control of Autonomous System, pp. 49 – 55, 2009.

[12] Dorigo, M. StÜtzle, The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances, Technical Report IRIDIA-2000-32.

[13] Naqiv, Z.N. Matheru, H. Chadha, K., Review of Ant colony Optimization on Vehice Routing Problems and Introduction to Estimated-Based ACO, International Conference on Environment Science and Engineering IPCBEE, Vol. 8, pp. 161 – 166, 2011.

[14] Dorigo, M. Gambardella, L.M, Ant Colony System: A Cooperative Learning Approach to Travelling Salesman Problem, IEEE Transactions on Evolutionary Computation, Vol.1, 1997.

[15] Dorigo, M. Gambardella, L.M., Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics , 1996.

[16] Kui-Hong, P.; Yong-Jae, K.; Jong-Hwan, K.; Modular Q-Learning Based Multi-agent Cooperation for Robot Soccer. Journal of Robotics and Autonomous Systems 2001, 35, 109 – 122.

[17] Claes, R. Holvoet, T., Cooperative Ant Colony Optimization in Traffic Route Calculations,PAAMS, Advances in Soft Computing, Springer, Vol. 155, pp. 23 – 34, 2012.

[18] Wang, Y. W.de Silva, C. "A machine-Learning Approach to Multi-Robot Coordination," in Journal of Engineering Applications of Artificial Intelligence 2, pp. 470-484, 2008.