The University of York Department of Computer Science

Submitted in part fulfilment for the degree of BEng.

Assessing the Difficulty of a Generated Path-Planning Problem

Stephen Lewis Webb

April 5th 2016

Supervisor: Dr. Rob Alexander

Number of words = 0, as counted by `wc -w`.
This includes the body of the report, and Appendices **??**, **??**, but not **??**.

Abstract

Here is where the abstract goes.

Dedication.

Acknowledgements

Ack. Ack. Ack.

Contents

List of Figures

List of Tables

Part I

Preliminaries

1 Introduction

1.1 Motivation

Autonomous robotics are increasingly finding application in unpredictable, dynamic environments such as offices, road networks and shopping centres. Human oversight in these situations may be minimal or unlikely. Furthermore, the decisions taken by robots in these applications may well be safety critical - consider, for example, the actions taken by an autonomous car when an unseen pedestrian runs into the road ahead. Such robots must be carefully tested so that we are confident that they may be trusted to operate alone in these environments.

The problem of exploring and navigating the environment has received much work, producing well known methods such as SLAM. Emphasis is now moving towards testing such navigation algorithms so that we may be reasonably confident of the safety of both the robot platform and those in the environment around it. We must be confident that the robot will act safely in the situations it may find itself in during the course of its duties.

There is increasingly argument that traditional testing methods alone are not sufficient to prove the safety of an autonomous robot [ref Rob]. Consider, for example systems coverage - testing each system component at least once. This may not reveal faults in situations which are not addressed by any of the system components. Likewise, while we might test using requirements coverage, we cannot be certain that the stated requirements do not neglect to identify some unforeseen circumstance. Likewise, scenario coverage may have also neglected to include a scenario which such circumstances.

We must, in addition to traditional testing, aim to test the robot in as many diverse situations as possible. However, given any high level situation description - for example, navigating around a person in a room, we can generate a practically infinite set of test cases - every possible position of the person, every possible configuration of the furniture, every possible combination of every such variable. Clearly it is intractable to test every possible situation - instead we must aim to test a diverse

enough range of different situations to ensure we are confident in the algorithms and platform of the robot.

1.2 Background

1.3 Project Aims

While the generator is capable of evolving diverse maps, no attempt is made to assess the difficulty of navigating such maps. Doing so would allow the system to prioritise more difficult maps, covering more trivial situations implicitly.

This project therefore aims to improve the classification carried out, building upon Wei's generator. To do so the factors which cause a map to be difficult to navigate will be explored and identified. Following this, the most promising will be selected and a method of measuring each defined. This measure will be implemented into the generator. The classification taxonomy will be extended to include these new measures, allowing selection of maps to take them into account. Finally, the extended taxonomy will be evaluated against the original taxonomy.

It is hoped that these extensions will improve the set of maps generated, resulting in both more diverse and more difficult maps. By increasing diversity, we cover more situations and potentially uncover more faults. By increasing difficulty, we avoid testing trivial situations and improve confidence in the results of our testing - if difficult situations are successfully covered, we may be confident that the more trivial maps would also be successfully covered. As a result, the time and cost of testing an autonomous robot may be reduced whilst improving our confidence in it's ability to act safely without human oversight.

1.4 Document Structure

This project is divided into the following structure.

Chapter 2 reviews existing literature on the testing of autonomous robotics and description of Path Planning Problems, before focusing on the factors that make an environment difficult to navigate.
Chapter 3 describes the problem this project addresses, defining the task to solve and the set of requirements to be delivered.
Chapter 4 sets out the design and implementation of the simulator and

extensions made to the generator, including rationale for design decisions made during the project.
Chapter 5 evaluates the extensions made to the generator utilising a set of test agents and the simulator.
Chapter 6 presents conclusions drawn from the evaluation of the changes made to the generator.
Chapter 7 suggests future work and improvements to the generator.

1.5 Statement of Ethics

2 Literature Review

2.1 Introduction

This section begins with review existing literature on testing methods for autonomous robots and path planning problems. The difficulties faced by these methods and representations and the potential improvements identified and discussed. Following this, literature on the factors of difficulty used to test current robotics (for example in competition scenarios) is discussed and a likely candidate for the goals of this project identified. Finally, this candidate is investigated in greater detail.

2.2 Testing Autonomous Robots

Testing autonomous robotics is a very difficult task. The environment in which they are to operate is often complex and dynamic; in addition many of the domains to which autonomous robots are being applied are safety critical. An example of such an application is autonomous road vehicles, which face the challenges posed by the roads and fellow road users and must behave in a safe manner. To consider such a vehicle safe, we must have confidence that it is capable of acting in a manner which does not endanger itself or fellow road users in any situation it may find itself in. In [1], Umerson et al. tested each algorithm driving their autonomous vehicle offline in simulation or via data replay, before moving onto on-vehicle testing. Similarly, Google's autonomous vehicles have been tested for thousands of hours[2]. On-vehicle testing, however, presents the possibility that something could go wrong, potentially endangering the vehicle, its occupants, and other road users. We might test for confidence offline, like Umerson - but hand crafting each simulation map is time consuming and restricts the total range of testing which may be carried out. Furthermore, faults may not become apparent until the system as a whole is tested on-vehicle outside of simulation.

Indeed, there is increasingly argument that new approaches to testing are required to sufficiently cover the range of situations a robot may

encounter while working autonomously. In [3], Alexander contrasts his work focusing on coverage of a range of situations with other methods such as requirements and systems based testing. Consider requirements testing - while we may successfully pass tests against stated requirements, this potentially misses failure cases which the requirements missed. Likewise, systems testing cannot identify faults that arise as a result of an external factor. Alexander argues that we must, in addition to other forms of testing, also test over a wide range of situations (for example, navigation of a robot in a room), the factors of such a situation (furniture, people, etc. in the room) and possible combinations and configurations of such factors (layout of furniture, position of people..). The aim of such testing is not to test every possible situation arising from these factors but to cover a wide and diverse range - and in doing so convince ourselves that the robot under test is safe. Hence, a large number of diverse testing scenarios are desired.

2.3 Path Planning Problems

A Path Planning Problem (PPP) is, in its simplest form, a specification of an environment to navigate and a description of its features, such as the size of the environment, location of obstacles, and so on. Richer aspects such as 3D terrain and surface materials might also be included. It is upon this environment that the robot or algorithm may be tested. Such specifications may range in complexity and features. In addition, the PPP may also specify the initial configuration of the robot under test. The problem is then to successfully find the (optimal or near optimal) range of motions from the initial position to the goal position, avoiding obstacles. PPPs can be used to test autonomous vehicles, algorithms, industrial robotic arms and so on. Indeed, they are not limited to simulation - one could use the specification to set up an arena with physical robots and obstacles in which to test the robot.

PPPs may be used for traditional testing for a robot or algorithm. As an example, consider requirements-based testing of an industrial robot arm. If a requirement states that the arm must be able to pick up an object and move it to another location, one could produce PPPs describing a simple environment with the object and target location and the configuration of the robot arm (before picking up the item, after picking up the item, for example) and then simulate the robot to ensure it moves from each position to the goal.

As discussed above, a chief difficulty in producing PPPs to test robots is the cost and time required to develop a large number of diverse problems, particularly if we must hand-craft them. In [4], Ashlock puts forward a PPP generator for simple 2D grid problems, based upon a genetic algorithm. He evolves the produced PPPs to maximise three measures - minimum number of turns and advances moves required to solve the PPP, and the sum of the minimum turns and advances, as calculated by a dynamic programming algorithm (DPA). This DPA also implicitly discards uncompletable PPPs. This approach allows for a large number of PPPs to be evolved in short order.

However, as noted above, it is not enough to test just a large number of situations - if they are all too similar (consider situations in which the only difference is a slight rotation in a piece of furniture) then very little practical benefit is gained from the testing, even over a large number of such tests. Thus, Ashlock also seeks to provide diversity in the final collection of PPPs. He makes use of a neighbour joining taxonomy inspired by the taxonomic procedures seen in biology. This requires taxonomic characters - measurable, computable qualities describing a PPP - to be extracted. Ashlock makes use of the three fitness function characteristics, alongside the total number of obstacles in the PPP grid as his four taxonomic characters. Finally, using the UPGMA clustering method, these taxonomic characters produce a tree of PPPs where like problems cluster in branches. Thus, by selecting PPPs from many branches, ignoring closely related nodes, a diverse collection of PPPs may be produced. As noted by Wei, under Ashlock's simple taxonomy it is still necessary to observe the diversity of a set of PPPs manually as the PPPs are only separated into groups but the diversity in each group is unknown[5, chapter 8, p. 66].

In [5], Wei reimplements Ashlock's work and attempts to evaluate and improve upon it. Through his testing, Wei shoes that Ashlock's generator produces superior PPPs compared to those produced by a random PPP generator by simulating robots with varying levels of completeness in their path planning algorithms. In all cases, the test agents had a lower success rate on Wei's PPPs, showing that his PPPs are better suited to exposing their flaws. In addition, Wei provides optimisation to the generator by pruning PPPs in which the goal is unreachable during a mating event, ensuring only completable PPPs are generated. This prevents the genes of the uncompletable PPP entering the pool and improves the time taken to generate a large number of PPPs. Wei goes on to show that this change also improves the PPPs themselves, again through simulating

robots and comparing the success rates against Ashlock's PPPs.

Ashlock and Wei's work provide a way to select diverse PPPs via clustering and a method to quickly produce a large number of such PPPs. However, neither attempt is made to measure how difficult these PPPs are to traverse - while a robot with perfect information of the map will clearly complete and PPP with a reachable exit via the optimal route, Wei's work shows that less capable robots often fail and become stuck. Indeed, Ashlock's fitness function would maximally reward a completely canalised PPP where the obstacles force the robot along one path in a manner that maximises the number of turns and advances the robot must make. Clearly, such a map would be time consuming but not actually be difficult to traverse, as there is no opportunity to become lost - the only choices are to proceed or reverse. Thus, the faults of a less capable robot are not likely to be discovered.

By assessing the difficulty of a PPP and applying this measure both to the fitness function and as a taxonomic character, the PPP generator put forth by Ashlock and Wei could be improved. We could evolve for more difficult PPPs as well as clustering similar difficulty PPPs together, allowing the user to select both diverse and difficult PPPs to solve during testing. As a result, we could improve our confidence in a robot under test by testing increasingly difficult situations. In addition, by testing difficult situations we can be confident that the robot can behave appropriately in more trivial situations.

2.4 Factors of Difficulty

In order to begin assessing the difficulty of a PPP, we must first understand the factors that make an environment difficult to navigate. As autonomous robotics become more prevalent, there are an increasing number of attempts to provide standardised tests for both the algorithms and the platforms, as well as competitions environments from which we might draw some of these difficulty factors.

As an example, consider RoboCup Rescue [6], a competition in which teams of robots attempt to carry out various tasks in a disaster environment, ranging from location of survivors to fighting fires and others besides. The characteristics of the Rescue competition domain seek to simulate a disaster and challenge the teams; robotics must be able to work in a large heterogenous team, planning in the long term and collaborating emergently in a situation that demands sharp real-time response.

In addition, the environment is potentially hostile due to debris and fire and information is missing, partial or incorrect. As such, completion of goals often requires exploration and robust planning to react to the unknown and dynamic environment.

In [7] Jacoff et. al. approach their evaluation of difficulty from a different angle to RoboCup; rather than a simulated situation they identify "fundamental elements of mobile robot autonomy" and seek to challenge these elements. These elements are locomotion, sensory perception, knowledge representation, autonomy from the operator, and collaboration. As such, they designed three arenas of increasing difficulty. Challenges include mazes, locomotion hazards, debris, sensory confusion from the environment and dynamic environments which alter explored areas of the map.

Ashlock has also worked on procedural generation of maze-like maps, taking a similar genetic approach to that used by his PPP generator [8]. Here, his difficulty factors and fitness functions are the shortest path from entrance to exit, accessibility of 'checkpoints' (simply points in the maze which are used to characterise connectivity and path length) in the maze, number of distinct branching paths, and number and length of dead ends.

We must take into consideration the range of robots and applications. It is not necessary to test all autonomous robots with the same rigour. While an autonomous road vehicle would need in-depth safety testing, simpler applications such as the Roomba household cleaning robot have more relaxed safety expectations upon them. In [9], Jacoff notes that a good testing method must ensure that the "figurative measuring stick is long enough to capture performance at both ends of the available robotic capabilities spectrum, and that it separates performance results in between." Therefore, the chosen measure of difficulty must be robust and adjustable so that it might be applied across a wide range of robots.

2.5 Exploration

A recurring theme in the literature discussed in the above section is that of exploration; particularly exploration in situations where information is unavailable or incorrect.Indeed, Wei's paper showed that robots which did not have prior knowledge of the PPP often struggled to reach the goal. This section will therefore focus on the problem of exploring an unknown map in order to develop a measurement of exploration which

might be applied to improve the generation of PPPs by allowing for evolution of 'hard to explore' PPPs.

In [10], Meyer and Filiat argue that path planning calls upon three processes - map learning, localization, and finally planning. Learning requires acquisition and memorization of data via exploration. Localization requires the derivation of the robot's current position, interdependent on learning - one must explore to know the current position, and must know the current position to know where to explore. Finally, path planning (as previously discussed) requires the choice of the course of actions to reach a goal from the current position. This requires the map to be represented in some way that is plannable.

The approach to representation of the robot's environment can be broadly decomposed into two categories - Topographical maps use the characterization of various places in the map by external sensors, storing their relative positions. This approach makes updating the map easier than other approaches as new information arrives and the map is reconsidered. However, localization is often much more difficult and highly sensitive - different locations, viewed from different angles or sensor configurations, may seem very similar and thus difficult to distinguish [10].

The second category is metric maps in which the position of obstacles is stored in a common reference frame (for example by decomposing the world into a grid and marking obstacle grid squares as impassable). A popular approach to such maps is the Occupancy Grid [11]. In this approach, the world is decomposed into a grid matrix and the probability that a given cell is occupied is calculated via sensor readings. A background uncertainty initially fills the map; cells with probabilities near this uncertainty therefore have little or no information gathered upon them. This naturally guides exploration; to explore one must plan to move into an area where the probabilities of the surrounding cells are close to the background uncertainty of the environment.

There are two approaches to updating a metric map; a backward model wherein the probability of a cell being occupied is calculated via sensor readings, and a forward model where the calculated probability of a sensor reading informs the map that would produce such a reading. There is some evidence to suggest the forward model produces a better map, though it is more sensitive to change in the environment and is prohibitively expensive for real time application [12].

Given a representation of the environment, there are many approaches to deciding where to move and explore. In [13], Lee describes several

such approaches. A reactive approach holds little internal state, reacting quickly to incoming sensor information, and is suitable for simple exploration methods such as wall following. Another approach is to simply go to the least explored region. A further approach is to "go where it's interesting", basing decisions on the latest map. This attempts to systematically reduce uncertainty in the map, often using the occupancy grid as a method of describing uncertainty. Under this representation, an 'interesting' area is an area with occupation uncertainty close to the background uncertainty - these regions are unexplored, or have very little data gathered upon them. The robot would therefore path to these areas as the pay-off in information to be gathered is higher.

A further, well investigated approach is SLAM - Simultaneous Localization and Mapping - the problem of placing a robot in an unknown environment and having it produce a consistent map whilst simultaneously locating itself within said map[14, p. 99]. There are a large number of approaches to SLAM, a few of which are open source [14, p. 107]. SLAM is, however, computationally expensive and complicated and therefore likely to be too heavyweight for the purpose of this project.

No approach is without its difficulties, however: one of the most common and most important difficulties faced by exploration and in mobile robotics in general is that of sensor fusion. It is rarely sufficient to explore with only a single sensor; this restricts the 'field of view' in which knowledge on the environment may be gathered and is not robust to difficulties to the sensor in the environment (for example, a laser sensor may struggle with a plate glass door). As such, multiple, varied sensors are used. Sensor fusion is the problem of fusing sensor readings from multiple independent sensors in order to produce coherent knowledge about the external environment. There are many approaches to tackling this; the best known are Dempster-Shafer Theory (DST) and Dezert-Smarandache Theory (DSmT). There is a range of literature dealing with the problem of sensor fusion and applying fused sensor readings to map representations. Another difficulty is communication of the explored map to human operators and fellow robots - In [7], Jacoff et. al. identified such communication as a fundamental element of robot autonomy, requiring that the robot is capable of expressing its maps understandably to the humans.

An additional difficulty is a direct algorithmic result of an uncertain environment - path planning approaches need to be dynamic and capable of adjusting to new information. For example, a path generated from what a priori knowledge is available may, during traversal and explora-

tion, be discovered to be blocked. The system must then adjust to this new knowledge and plan a new path. This difficulty requires the path planning algorithm to be coupled with robot movement such that the search space may be updated and costs reconsidered if it is necessary to take a new obstacle into account.

Some approaches to this problem plan a global path based on a priori knowledge and then attempt to path around discovered obstacles, recomputing the global path if necessary. Other approaches generate a route which seems optimal based on current knowledge and proceeds upon it until the sensors detect discrepancies; a new route is then generated and taken [15]. If a priori knowledge is not available, it may be the case that the optimal route is not obvious until the map is first suitably explored. Finding an optimal solution to PPPs without perfect a priori knowledge is therefore a harder problem, though the robot may benefit in the future from its exploration if the environment is unchanging.

As discussed above, this chosen difficulty factor must be both robust and adjustable so that it might be applied across the range of autonomous robots. By evolving PPPs which favour high exploration difficulty - causing, for example, more obstacles to be placed and in a manner which challenges the exploring agent with dead-ends and multiple choices of routes, we can test capabilities at the far end of the spectrum. Similarly, the fitness function might be adjusted to encourage less difficult PPPs to test the lowest end of the spectrum and the range in between.

Part II

Research

3 Problem Analysis

3.1 Introduction

Testing of autonomous robots is a difficult task and, as they become more commonly applied to real world problems, it is increasingly important that we are confident in their safe behaviour. As demonstrated by the literature review, this requires that we cover as many situations that may be encountered in the robot's work as possible - both those we can predict, and those that may not be obvious beforehand. While hand-crafted test situations provide insight into such behaviour, and can be carefully tailored to challenge weak areas of the algorithms, robot platform, and so on, they are time consuming to create and have no hope of catching failures that we can't predict (and therefore can't provide a scenario for) at a testing stage.

Previous work by Ashlock and Wei provides us with the means to algorithmically generate large numbers of diverse path planning problems as test cases, taking us some way in addressing the issue of testing a wide range of diverse and potentially unexpected situations. However, the simple fitness function employed in generating these PPPs would maximally reward an environment where the test agent is forced along a canalised, winding path which maximises distance travelled and turns made, with no opportunity for deviation. Clearly, such an environment is not a challenging test case - the only choice is between proceeding or reversing.

Further work is required to improve the generated problems such that the difficulty of a generated PPP can be assessed more accurately and evolved for as part of the fitness function. This would allow for selection of diverse test cases of a difficulty suitable to the application of the robot under test. The literature review identifies a range of hazards and difficulties employed by current testing standards and robot competitions; the chief amongst these is exploring an unknown map and successfully identifying the optimal route.

This section begins with a high level definition of the goal and task of this project, defining the terms and measurements discussed above in

more detail. Subsequently, the high level requirements will be laid out and rationalised.

3.2 Task Definition

The high level goal of this project is to improve Wei's PPP generator to assess and evolve for difficulty in the output path planning problems. These path planning problems are then to be used to find faults in test robots; hence the improved generator should be more capable at identifying faults by causing more test robots to fail to reach the goal location.

3.3 Difficulty

As demonstrated in the literature review, there are numerous ways in which a robot test environment may be said to be difficult. These include but are not limited to locomotion hazards such as rubble, complex collaboration tasks and knowledge representation issues. Currently the PPPs produced by Wei's generator are simple 2D grid environments and do not include such complex tasks or rough but passable terrain. However, a common theme in all of the competitions and standard tests discussed is that of knowledge representation and exploration. Clearly, no matter how full the environment is with debris or sensory hazards, any robot with perfect prior knowledge of the PPP will be able to take the optimal route to the goal. The difficulty actually lies in exploration of the environment, identifying these hazards and planning how best to avoid or handle them such that the best route can be found. Therefore, the measure of difficulty will seek to assess how difficult it is to explore a given PPP.

3.3.1 Occupancy Grid

Wei's generated PPPs can be considered as a 2D grid with a defined start and goal position and certain cells occupied by obstacles as described by the PPP itself. A test agent with some sensory capability - for example, perfect 'sight' into immediately surrounding cells - must then explore and path to the goal. A dynamic programming algorithm (DPA) is used to assess the optimal path across this grid.

As discussed in the literature review, the occupancy grid map representation closely mirrors this representation. Exploration progress of a bounded map may be understood by initially setting the entire grid to a background 'noise' occupation certainty. Then, during exploration, visited areas of the map become are updated and their occupation becomes more certain. Thus unexplored areas are close to the background uncertainty.

3.3.2 Difficult PPPs

A 'difficult' PPP is one where the goal position and optimal route from the initial position is difficult to identify in full without exploration. As such, the actual path taken may be far from optimal and the optimal route may only be apparent once the goal position has already been found and reached. The test agent, as a result of this deviation from the optimal path, incurs a higher than optimal movement cost to reach the goal position.

3.3.3 Measurement of Difficulty

By applying the occupancy grid map representation to the current PPP representation, uncertainty may be included in the DPA assessment of the optimal path from the initial position to the goal. Adjustments to the DPA evaluation function can be made to take into account the reward of exploration - finding parts of the optimal route, or the goal position - alongside the cost of movement. Hence, the DPA may model exploration of the PPP from very little a priori knowledge by a simple agent. The resulting optimal route produced by the DPA may be different from the optimal route as assessed by movement cost with perfect a priori knowledge - the deviation may hence be computed and the cost incurred by the requirement to explore derived. A more difficult PPP will therefore require more exploration and make the route difficult to find, increasing this cost. As a result, larger PPPs and those with a greater number of obstacles will clearly be measured as more difficult than smaller, simpler PPPs.

3.3.4 Application and Diversity

The difficulty measurement can be included in the fitness function and as a taxonomic character. Inclusion in the fitness function allows the

Requirement	Description
R01	The generator must assess a notion of difficulty (as defined above) of a PPP.
R02	The difficulty of produced PPPs must be adjustable.
R03	The generator must be capable of producing diverse PPPs.
R04	The measurement of difficulty must be included in the clustering taxonomy.
R04a	No single taxonomic character may dominate the taxonomy.

Table 3.1: Generator Requirements

difficulty of generated PPPs to be adjusted and tailored to a level suitable for the robot under test. In addition, by using this measure as a taxonomic character, the produced PPPs will be clustered into groups of similar difficulty. Hence the final selection of test problems should be improved in difficulty while still allowing for diversity. This should improve the capability of generated PPPs to identify faults in test robots.

3.4 Generator Requirements

3.4.1 Rationale

The overall aim of this project is to assess a measurement of the difficulty of a PPP. R01 guarantees this. As discussed previously, a good testing standard should be able to challenge the whole spectrum of capabilities. This is covered by R02. We must also test for diverse PPPs, hence R03. The generator currently assesses diversity via a clustering taxonomy. The new measure must be included in this taxonomy so that it might be considered during PPP selection, hence R04.

3.5 Simulator Requirements

The code for Wei's evaluation simulator is not available, hence a new simulator must be developed to evaluate the changes made to the generator.

Requirement	Description
S01	The simulator must accept and simulate the generated PPP format.
S02	Agents with a variety of capabilities must available for testing.
S03	Simulation results must be recorded to disk.
S04	Simulation must terminate in a finite time.

Table 3.2: Simulator Requirements

3.5.1 Rationale

Wei's work laid out a .ppp file format for generated PPPs. S01 ensures the simulator tool can read these independent of changes to the generator. To gather meaningful results, tests must be carried out over a range of agent capabilities, as set out by S02. In particular, buggy agents with known faults must be available so that the ability of the improved PPPs to detect such faults may be understood in comparison to Wei's PPPs. To allow for further analysis, these results must be stored as set out in S03. As we are dealing with agents with faults, it may be the case that an agent may path in a loop and never reach the exit. In this case the simulator should record the test run as a failure for that agent, as never reaching the goal is a clear fault. This is set down by S04.

4 Design and Implementation

5 Evaluation

Part III

Closing Remarks

6 Conclusion

7 Future Work

8 Bibliography

[1] C. Umerson et al., "Autonomous Driving in Urban Environments: Boss and the Urban Challenge", J. Field Robotics Special Issue on the 2007 DARPA Urban Challenge, vol. 1 no. 8 pp. 425-466 Jun. 2008

[2] E. Guizzo. (2011, Oct. 18). How Googles Self Driving Car Works [Online]. Available: http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works,

[3] R. Alexander et al., "Situation coverage - a coverage criterion for testing autonomous robots", Dept. Computer Science, University of York, Report Number YCS-2015-496, Apr. 2015

[4] D. A. Ashlock et al., "Evolving A Diverse Collection of Robot Path Planning Problems" in IEEE Congress on Evolutionary Computation, Vancouver, BC 2006, pp. 1837 - 1844

[5] H. Wei, "Evolving path-planning problems to find faults in autonomous mobile robots", MSc thesis, Dept. Computer Science, University of York, York, N.Yorkshire, 2013

[6] H.Kitano, S. Tadokoro, "RoboCup Rescue: A Grand Challenge for Multiagent and Intelligent Systems", AI Magazine, Vol. 22 No. 1, pp. 39-52, Spring 2001

[7] A. Jacoff et al., "Performance Evaluation of Autonomous Mobile Robotics", Int. J. Industrial Robot, Vol. 29, No. 3, pp. 256-267, 2002

[8] D.A. Ashlock et al., emph"Search-based Procedural Generation of Maze-Like levels", IEEE. Trans. Comput. Intelligence and AI in Games, vol. 3 no. 3 pp 260-273 Sept. 2011

[9] Standard Test Mestods for Response Robots, ASTM International Standards Committee on Homeland Security Applications (E54.08.01), 2015

[10] J. Meyer, D. Filliat, "Map-based navigation in mobile robots:: II. A review of map-learning and path- planning strategies", Cognitive Systems Research, Vol. 4 no. 4, pp. 283-317 Dec. 2003

[11] A. Elfes, "Using Occupancy Grids for Mobile Robot Perception and Navigation" Computer, Vol. 22, no. 6, pp. 46âĂŞ57, Jun. 1989.

[12] S. Thrun, "Learning Occupancy Grid Maps with Forward Sensor Models", Autonomous Robots, Vol. 15 no.2, pp. 111âĂŞ127, Sept. 2003

[13] The Map Building and Exploration Strategies of a Simple Sonar-Equipped Mobile Robot D. Lee Cambridge University Press, 1996

[14] H. Durrant-Whyte, T. Bailey, "Simultaneous Localization and Mapping: part 1", IEEE Robot. Automat. Mag., Vol. 13, No. 2, pp. 99-110, Jun. 2006

[15] A. Stentz, "Optimal and Efficient Path Planning for Partially-Known Environments", Proc. IEEE Robotics and Automation, San Diego, CA, May 1994